ANNUAL STATUS REPORT

Error Control Techniques for Satellite and Space Communications

NASA Grant Number NAG5-557

Principal Investigator:
Daniel J. Costello, Jr.

October 1991

Summary of Progress

During the period February 1, 1991 - July 31, 1991, progress was made in the following areas:

1) Coding Gains for Bandwidth Efficient Codes

With his 1948 paper "*The Mathematical Theory of Communication*" Claude E. Shannon stimulated a body of research that has evolved into the two modern fields of Information Theory and Communication Theory. That one paper should spawn two active research areas is extraordinary and, as will become apparent, a direct consequence of the nature of the results. The fundamental philosophical contribution of this seminal treatise was the formal application of probability theory to the study and analysis of communication systems. The theoretical contribution of Shannon's work was a useful definition of "information" and several "channel coding theorems" which gave explicit upper bounds, called the channel capacity, on the rate at which "information" could be transmitted reliably on a given communications channel.

In the context of current research in coded modulation, the result of primary interest is the "noisy channel coding theorem for continuous channels with average power limitations." This theorem states that the capacity C of a continuous additive white Gaussian noise (AWGN) channel with bandwidth B is given by

$$C = B \log_2 \left(1 + \frac{E_s}{N_0}\right) \tag{1}$$

where E_s is the average signal energy in each signalling interval T, and $N_0/2$ is the two sided noise power spectral density. This theorem is both profound in its implications and, fortunately for communication engineers, frustrating in its ambiguity.

It is profound, because it states unequivocally that for any transmission rate, R, less than or equal to the channel capacity C, there exists a coding scheme that achieves an arbitrarily small probability of error; conversely, if R is greater than C, no coding scheme can achieve reliable communication. The field of Information Theory is, in a strict sense, an effort to apply Shannon's definition of information and methods of analysis to different channels and problems, such as cyptography. It is frustrating, because like most existence theorems it gives no hint as to how to find the appropriate coding scheme or how complex it must be. Communication engineers and coding theorists make their living trying to create schemes that achieve the levels of performance promised by Shannon's results. Figure 1 is both a measure of how close they have come and how much better they can possibly do.

The bound of equation (1) can be put into a form more useful for the present discussion by introducing the parameter K to represent the average number of information bits transmitted per signalling interval. Assuming perfect Nyquist signalling, then

$$0 \leq K \leq C/B$$

and

$$E_s/N_0 = KE_b/N_0,$$

where E_b is the average energy per information bit. Substituting the above relations into equation (1) and performing some minor manipulations yields

$$E_b/N_0 \geq \frac{2^K - 1}{K}, \tag{2}$$

which relates the bandwidth efficiency K to the signal-to-noise ratio (SNR) E_b/N_0. This bound, labelled Shannon's Bound, is plotted in Figure 1 and represents the absolute best performance possible for a communications system on the AWGN channel.

In this form, Shannon's bound gives the minimum signal-to-noise ratio required to achieve a specific bandwidth efficiency with an arbitrarily small probability of error. For example, if one wants to transmit $K = 1$ information bits per channel symbol (signalling interval), then there exists a coding scheme that operates reliably with an SNR of $0dB$. Conversely, any coding scheme, no matter how complex, sending $K = 1$ information bits per symbol with an SNR *less than* $0dB$ will be unreliable. The bound of equation (2) also manifests the fundamental tradeoff between bandwidth efficiency and SNR. That is, increased bandwidth efficiency can be reliably achieved only with a corresponding increase in minimum SNR. At this point, it is important to reiterate that Shannon's results do not suggest what code or what type of signalling is necessary to achieve this bound, and consequently it can be a discouraging measure of a system's performance.

In real communication systems, there are many practical considerations that take precedence over Shannon's bound in design decisions. For example, satellite communication systems that use nonlinear travelling wave tube amplifiers (TWTA's) require constant envelope signalling such as M-ary phase shift keying (MPSK). Thus, even if Shannon's results firmly stated that capacity at a bandwidth efficiency of $K = 3$ information bits per symbol can be achieved with a rate 3/4, 256 state, convolutional code using 16 quadrature amplitude modulation (QAM), it would not be feasible to do so on the TWTA satellite link.

It therefore seems reasonable to ask what the minimum SNR required to achieve reliable communication is *given* a modulation scheme and a bandwidth efficiency, K. For the discrete input, continous output, memoryless AWGN channel with M-ary one dimensional, e.g., amplitude modulation (AM), or two dimensional (PSK, QAM) modulation and assuming equiprobable signalling, the capacity bound becomes

$$K^* = \log_2(M) - \frac{1}{M}\sum_{i=0}^{M-1} E\left\{\log_2 \sum_{j=0}^{M-1} \exp\left[\frac{|a^i + n - a^j|^2 - |n|^2}{N_0}\right]\right\}, \tag{3}$$

where a^i is a modulator symbol, n is a Gaussian distributed noise random variable with mean 0 and variance $N_0/2$, M is the number of modulation symbols, and E is the expectation operator. The bound of equation (3) is plotted in Figure 1 for BPSK, QPSK, and 8PSK modulation.

For a specified signalling method and bandwidth efficiency, this bound represents the minimum SNR required to achieve reliable communication. For example, to send $K = 1$ information bits per signalling interval using QPSK modulation requires a minimum SNR of $E_b/N_0 = 0.5dB$. Any system using QPSK modulation with $K = 1$ and operating with a SNR lower than $0.5dB$ will not be reliable, regardless of complexity.

Also depicted on the figure is the performance of a number of real coded communications systems with a variety of bandwidth efficiencies. These points are plotted by determining, either analytically or by simulation, the SNR required for the system to achieve an information bit error rate (BER) of 10^{-5}. (Thus, a BER of 10^{-5} is chosen as the "arbitrarily small probability of error.") By comparing these points to the corresponding bound with the same bandwidth efficiency and type of modulation, it can be seen how close a system is to the ultimate performance. For example, the well known rate $R = 1/2$, memory 6, convolutional code sends $K = 1$ information bits per QPSK symbol with a BER of 10^{-5} at an SNR of $E_b/N_0 = 4.4dB$. This is $3.9dB$ away from the QPSK bound and $4.4dB$ away from Shannon's bound.

The performance of a number of recent trellis coded modulation (TCM) schemes are also shown on the figure. For an information rate of 2 bits per symbol, the Ungerboeck $R = 2/3$, memory 6, 8PSK trellis code is $3.0dB$ from the bound and performs $0.4dB$ better than the $R = 2/3$, memory 6, 8PSK pragmatic trellis code suggested by Viterbi. It should be noted that the previous comment reflects performance at a BER of 10^{-5}; the Ungerboeck code has an asymptotic coding gain of $5.0dB$ compared to $3.0dB$ for the pragmatic code.

To achieve an information rate of 3 bits per symbol with constant envelope signalling, 16PSK can be used. The best known $R = 3/4$, memory 6, 16PSK trellis code achieves a BER of 10^{-5} with an SNR of $E_b/N_0 = 9.6dB$ and, as shown, is about $6.0dB$ from the Shannon Bound. If constant envelope signalling is not required, then quadrature amplitude modulation (QAM) offers improved performance at high information rates, i.e., more bits per symbol. The performance of three $R = 3/4$, 16QAM, trellis codes are shown in the figure. The memory 4, 16QAM convolutional code proposed by TRW performs $0.5dB$ better than the 16PSK code even though it has fewer states. Further improvement is available if 16QAM TCM is used. A linear, memory 4, 16QAM trellis code is $1.1dB$ better than the 16PSK code and the nonlinear, memory 6, 16QAM code is $1.8dB$ better. The latter code also has the advantage of being fully rotationally invariant.

Recent advances in coding theory, including coded modulation and constellation shaping, and the technological feasibility of increasingly complex coding schemes have brought the bounds of Shannon and other information theorists within sight. In fact, it has been suggested that with sophisticated shaping techniques, complex codes, and large lattice theoretic constellations capacity may be achieved in some specialized systems in the near future. Figure 1 illustrates the progress made toward that goal.

2) Hardware Implementation of a Bandwidth Efficient Coding Scheme for the Hubble Space Telescope

As a demonstration of the performance capabilities of trellis codes using multidimensional signal sets, a Viterbi decoder was designed and implemented for a 16-state, rate 5/6, 2.5 bits/symbol, 4-dimensional 8PSK trellis code. The choice of code was based on two factors.

The first factor was its application as a possible replacement for the coding scheme currently used on the Hubble Space Telescope (HST). The HST at present uses a rate 1/3, $v = 6$ convolutional code (with $2^v = 64$ states) with BPSK modulation. With the modulator restricted to 3M symbols/second this implies a data rate of only 1M bits/second, since the bandwidth efficiency $K = 1/3$ bit/symbol. This is a very bandwidth inefficient coding scheme, although it has the advantage of simplicity and large coding gain.

The basic NASA requirement was for a scheme that has as large a K as possible. Since a satellite channel was being used, 8PSK modulation was selected. This allows a K of between 2 and 3 bits/symbol. Another influencing factor was INTELSAT's intention of transmitting the SONET 155.52 M bits/second standard data rate over the 72 MHz transponders on its satellites. This requires a bandwidth efficiency of around K = 2.5 bits/symbol. A Reed-Solomon block code is used as an outer code to give a very low bit error rate (BER).

The 16-state, 2.5 bits/symbol code chosen for implementation has reasonable complexity and a coding gain of 4.8 dB compared to uncoded 8PSK. This code also has the advantage that it is 45° rotationally invariant. This means that the decoder can synchronize to any one of the eight PSK symbols and still recover the data.

A paper describing the operation of the decoder will be presented at the NASA Lewis Research Center 2^{nd} Space Communications Technology Conference [1]. A copy of the paper is included as Appendix A of this report. The actual hardware decoder was delivered to the NASA Center for Space Telemetering and Telecommunications Systems at New Mexico State University in August. Its operation will be demonstrated as part of the NASA Lewis Conference in November.

3) Construction of Long Trellis Codes for Use With Sequential Decoding

The construction of good large constraint length trellis codes for use with sequential decoding is one of the primary objectives of this research. We have made progress on this problem in two directions. Because of the variable computational requirements of sequential decoding, codes with a rapidly growing column distance function (a good distance profile) must be chosen. This reduces the time required to find the correct path and thus minimizes the probability of buffer overflow. A large minimum free Euclidean distance (d_{free}) is also needed to minimize the probability of following an incorrect path. In [2], we have constructed systematic feedforward rate 2/3 trellis codes for 8-PSK modulation with constraint lengths up to $v = 25$ and rate 3/4 trellis codes for 16-QAM modulation with constraint lengths up to $v = 15$. These codes have both a good distance profile and a large free distance and can achieve asymptotic coding gains up to 6.53 dB over uncoded modulation. Hence they are well suited for use with sequential decoding. A copy of [2] is included as Appendix B of this report.

In [3], we have taken a different approach to the construction of good long codes. Since the determination of the distance properties of long codes is a very time-consuming task, and since sequential decoding is a very fast algorithm when the channel signal-to-noise ratio (SNR) exceeds the channel cutoff rate, we have constructed new long codes by generating codes at random and then testing their performance with sequential decoding. We have found that by testing about 100 randomly selected codes, some very good codes can be found. That this approach is successful can be explained by the well known fact of information theory that a randomly selected code is with high probability a good code, i.e., better than average. Hence a relatively small sample of codes will almost certainly contain at least one very good code. As an example, consider the construction of systematic feedback rate 2/3, constraint length $v = 8$ trellis codes for 8-PSK modulation. The number of possible codes is $2^{21} \approx 2 \times 10^6$. Instead of attempting an exhaustive search for the best code, we randomly chose 100 codes and evaluated their performance at an SNR of 8.0 dB using sequential decoding. The code with the lowest BER performs just as well as the best previously known $v = 8$ code over a wide range of SNR's. It is surprising to note that this code was found by examining the performance of less than a fraction of 5×10^{-5} of all possible codes. A copy of [3] is included as Appendix C of this report.

4) Performance Analysis of Multi-Level Trellis Codes

Multi-level trellis coding is a form of trellis coded modulation (TCM) which allows several different convolutional codes to interact in the selection of a signal point for transmission. In a single-level (Ungerboeck) TCM scheme, a single convolutional code of rate k_c/n_c along with k_u uncoded information bits are used to select a signal from a constellation of size $2^{n_c+k_u}$. In a multi-level TCM scheme, one output bit from each of m component codes along with k_u uncoded information bits are used to select a signal from a constellation of size 2^{m+k_u}.

The main advantage of multi-level TCM over single-level TCM is that a simplified multi-stage decoding technique can considerably reduce the complexity needed to achieve a desired coding gain. As such, it offers the possibility of better performance for a given complexity, or less complexity for a given performance, than single-level TCM.

The performance of multi-level trellis codes with multi-stage decoding has been studied and compared to the performance of single-level (Ungerboeck) trellis codes with Viterbi decoding using computer simulation [8]. The goal was to determine which scheme has the best performance for a given decoding complexity in terms of real coding gain at a BER of 10^{-5}. The simulation results indicate that the multi-level codes do not have a significant performance/complexity advantage over the Ungerboeck codes at moderate BER's. This can be attributed to two factors: the suboptimum nature of multi-stage decoding and the dense distance spectrum (high path multiplicity) of multi-level codes. However, since the effect of a high path multiplicity is less significant at lower BER's, the performance/complexity advantage is expected to shift to multi-level codes at BER's below 10^{-5}. A copy of [8] is included as Appendix D of this report.

5) M-Algorithm Decoding of Trellis Codes

The M-algorithm is a reduced complexity version of the Viterbi algorithm that achieves suboptimum error performance. It can be considered as an alternative to sequential decoding for achieving very low BER's with large constraint length trellis codes. It can also be viewed as a reduced complexity alternative to Viterbi decoding for short constraint length trellis codes. In particular, we have studied the performance of the M-algorithm in decoding a new nonlinear, 90° rotationally invariant, rate 3/4, 64 state, 16-QAM code with a 5.44 dB asymptotic coding gain. This new code transmits 3 bits/symbol and achieves almost 5 dB real coding gain at a BER of 10^{-5} over uncoded 8PSK. It is being considered for adoption by CCITT as the V.FAST coding standard for Two-Wire High-Speed Modems.

One problem with a practical implementation of this code is that with $\tilde{m} = 2$ coded information bits and constraint length $v = 6$, the trellis complexity $2^{\tilde{m}+v} = 2^8 = 256$ is quite large. This may be acceptable for users who require the full 5 dB coding gain. However, other users who will not require as much coding gain may wish to employ a decoder with less complexity. It is with this application in mind that we investigated the performance of the proposed code with reduced complexity M-algorithm decoding.

Our results show that with decoding complexity equivalent to a 16 state code, the performance of the new 64 state code with the M-algorithm is as good as the best known 16 state code with Viterbi decoding at a BER of 10^{-5}. When compared to Viterbi decoding of the new 64 state code, the M-algorithm with 16 states loses less than 0.5 dB at 10^{-5} and approaches the performance of Viterbi decoding asymptotically. Hence we conclude that the M-algorithm is a promising candidate for suboptimum, reduced complexity decoding of the proposed new 64 state CCITT V.FAST code. A paper summarizing our work on M-algorithm decoding of the proposed new code is being prepared for submission to the *IEEE Transactions on Communications* [5]. A preliminary draft of this paper is included as Appendix E of this report.

References

[1] S. S. Pietrobon and D. J. Costello, Jr., "A Bandwidth Efficient Coding Scheme for the Hubble Space Telescope", to be presented at the Second Space Communications Technology Conference, Cleveland, Ohio, November 1991.

[2] S. S. Malladi, F. Q. Wang, D. J. Costello, Jr., and H. C. Ferreira, "Some Systematic Trellis Codes with a Good Distance Profile", submitted to the IEEE Transactions on Communications, July 1991.

[3] F. Q. Wang and D. J. Costello, Jr., "Probabilistic Construction of Trellis Codes", *Proc. of the IEEE International Symposium on Information Theory*, p. 200, Budapest, Hungary, June 1991.

[4] J. Wu, D. J. Costello, Jr., and L. C. Perez, "On Multi-Level Trellis MPSK Codes", *Proc. of the IEEE International Symposium on Information Theory*, p. 145, Budapest, Hungary, June 1991.

[5] L. C. Perez and D. J. Costello, Jr., "M-Algorithm Decoding of the Proposed Nonlinear 64-State V.FAST Code", to be submitted to the *IEEE Transactions on Communications*, December 1991.

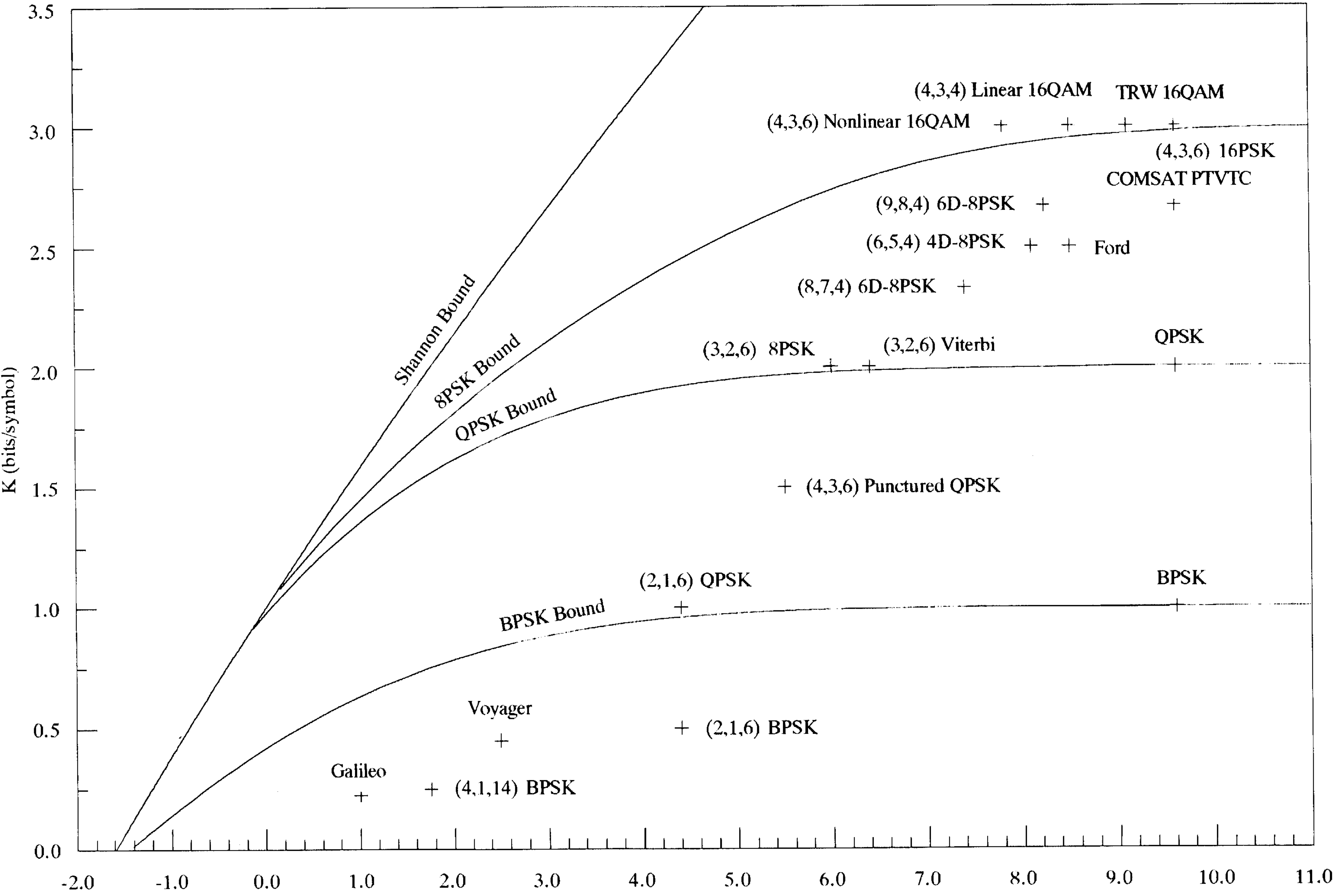


Figure 1: Plot of Efficiency, K (bits/symbol), versus E_b/N_0 (dB) for a BER of 1×10^{-5}

ORIGINAL PAGE
COLOR PHOTOGRAPH

Appendix A

A Bandwidth Efficient Coding Scheme for the Hubble Space Telescope

A Bandwidth Efficient Coding Scheme for the Hubble Space Telescope*

Steven S. Pietrobon
University of South Australia
The Levels, South Australia 5095
Australia

Daniel J. Costello, Jr.
University of Notre Dame
Notre Dame, Indiana 46556
U.S.A.

1 SUMMARY

As a demonstration of the performance capabilities of trellis codes using multidimensional signal sets, a Viterbi decoder for one of the codes in [1] was designed. The choice of code was based on two factors.

The first factor was its application as a possible replacement for the coding scheme currently used on the Hubble Space Telescope (HST). The HST at present uses the rate 1/3 $\nu = 6$ (with $2^{\nu} = 64$ states) convolutional code with BPSK modulation. With the modulator restricted to 3 Msym/s, this implies a data rate of only 1 Mbit/s, since the bandwidth efficiency $K = 1/3$ bit/sym. This is a very bandwidth inefficient scheme, although the system has the advantage of simplicity and large coding gain.

The basic requirement from NASA was for a scheme that has as large a K as possible. Since a satellite channel was being used, 8PSK modulation was selected. This allows a K of between 2 and 3 bit/sym. The next influencing factor was INTELSAT's intention of transmitting the SONET 155.52 Mbit/s standard data rate over the 72 MHz transponders on its satellites. This requires a bandwidth efficiency of around 2.5 bit/sym. A Reed-Solomon block code is used as an outer code to give very low bit error rates (BER).

The 16 state rate 5/6, 2.5 bit/sym, 4D-8PSK trellis code from [1] was selected. This code has reasonable complexity and has a coding gain of 4.8 dB compared to uncoded 8PSK [2]. This trellis code also has the advantage that it is 45° rotationally invariant. This means that the decoder needs only to synchronise to one of the two naturally mapped 8PSK signals in the signal set.

2 ENCODER IMPLEMENTATION

At first, a systematic encoder was used in the design. However, it was found that in designing a Viterbi decoder, it would be simpler if a non-systematic convolutional encoder was used. This is because the state transitions in a non-systematic encoder are highly structured, compared with the almost "random" transitions of a systematic encoder.

To convert the systematic encoder to a non-systematic form, the technique described in [3] is used. This method uses the fact that the impulse response of each shift register in a non-systematic encoder will produce output sequences that are equivalent to the generator polynomials. Since a systematic encoder must also produce the same sequences, it is relatively easy to find $\tilde{k}$ linearly independent output sequences from a systematic encoder that

*This work was supported in part by NASA Grant NAG5-557 and in part by OTC Limited under Project 1662.

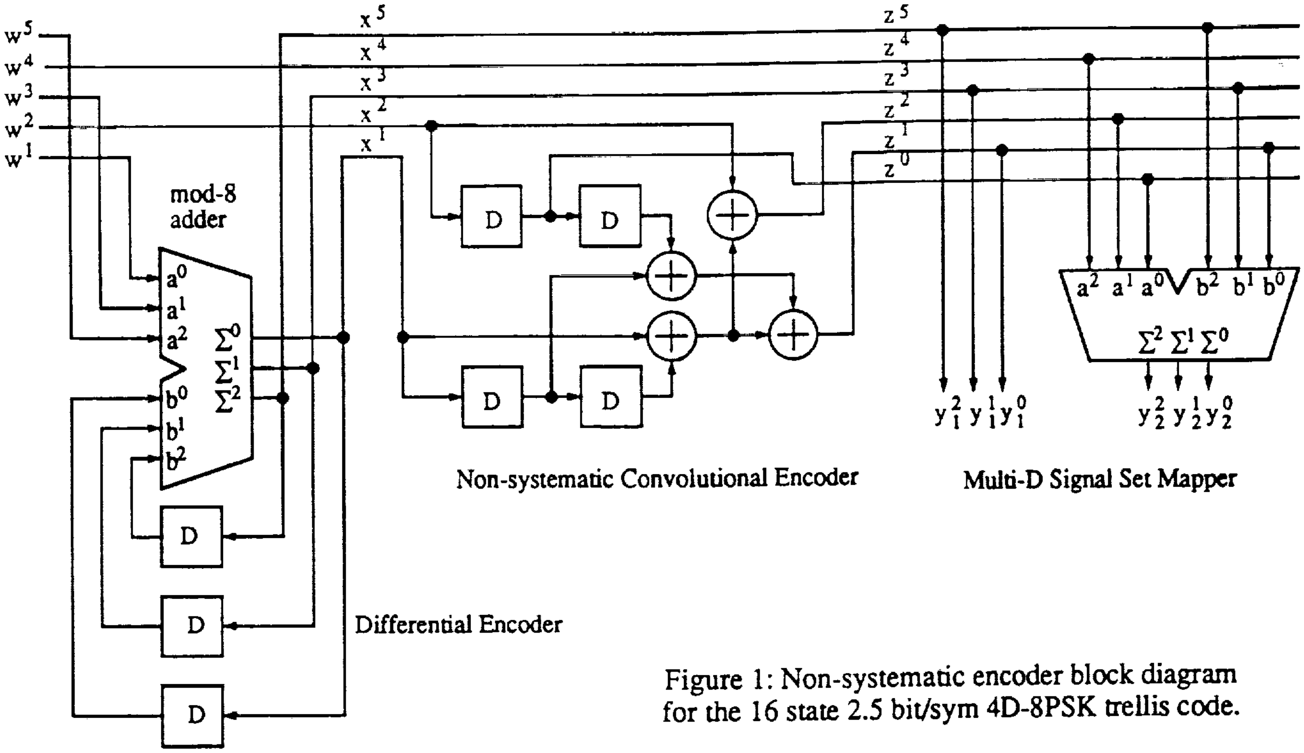


Figure 1: Non-systematic encoder block diagram for the 16 state 2.5 bit/sym 4D-8PSK trellis code.

can be used as generators of a non-systematic encoder.

There is usually more than one set of possible generator polynomials. The polynomials are chosen so that the inputs $x^1(D)$ and $x^2(D)$ are affected by a 45° phase rotation in the same way as in a systematic encoder. Thus, the differential encoder for the systematic code can also be used for the non-systematic encoder. The non-systematic encoder equations that were found for the 4D-8PSK code are

$$z^2(D) = x^2(D) \oplus (D^2 \oplus 1)x^1(D), \qquad (1a)$$

$$z^1(D) = D^2x^2(D) \oplus (D^2 \oplus D \oplus 1)x^1(D), \qquad (1b)$$

$$z^0(D) = Dx^2(D). \qquad (1c)$$

Figure 1 illustrates the new non-systematic encoder. After a 45° phase rotation, we have $z_r^2(D) = z^2(D)$, $z_r^1(D) = z^1(D) \oplus 1(D)$, and $z_r^0(D) = z^0(D)$. Rotating the equations in (1) gives $x_r^2(D) = x^2(D)$ and $x_r^1(D) = x^1(D) \oplus 1(D)$, the same as for the systematic encoder.

The encoder uses a Phase Locked Loop (PLL) to generate the two times clock for transmitting the two 2D symbols. This PLL is based on the 74HC4046 Integrated Circuit (IC). The encoder is able to accept data either serially or in five bit bytes.

3 DECODER IMPLEMENTATION

Due to the complexity of the decoder design, only a brief description is given here. As such, only the important design decisions are described.

To reduce the cost of the codec, a serial implementation of the decoder was chosen. That is, one clock cycle would be required for each state of the code. Since there are 16 states, at least 16 clock cycles are required to process each received 4D point. As will be described in more detail later, an extra seven clock cycles are required for start-up purposes. Thus, a total of 23 clock cycles are required for each iteration of the Viterbi algorithm.

The technology and clock speed in our design is the same as used in another Viterbi decoder designed by the author [4]. This gave us greater confidence that

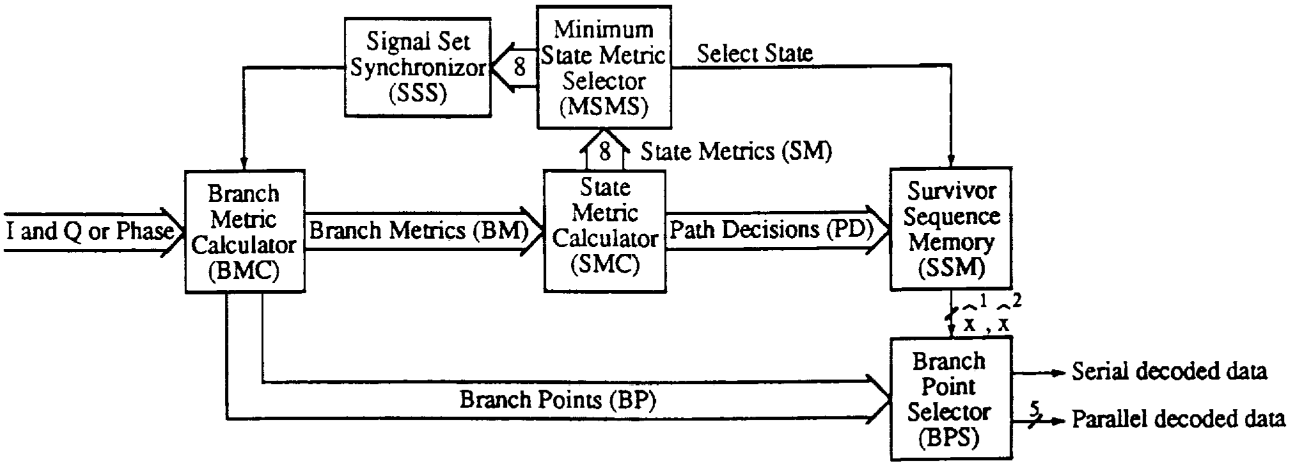

Signal Set Synchronizor (SSS)
8
Minimum State Metric Selector (MSMS)
Select State
8
State Metrics (SM)
I and Q or Phase
Branch Metric Calculator (BMC)
Branch Metrics (BM)
State Metric Calculator (SMC)
Path Decisions (PD)
Survivor Sequence Memory (SSM)
$\hat{x}^1, \hat{x}^2$
Branch Points (BP)
Branch Point Selector (BPS)
Serial decoded data
5
Parallel decoded data

Figure 2: Block diagram of a Viterbi decoder for the 16 state 2.5 bit/sym 4D-8PSK trellis code.

the design would work, even though the actual design is twice as complicated. Our design uses a 10 MHz clock (giving 100 ns clock cycles) and Schottky TTL logic for its ease of use and large variety of functions. The actual technologies used are 74LS (Low-power Schottky TTL) for non-time critical sections of the circuit and 74F (Advanced Schottky TTL) for time critical sections. Other technologies are used for functions not available in 74F or 74LS.

The decoder is operated asynchronously to the received data clock. This requires one of the seven extra clock cycles described above. Internally, the decoder operates synchronously to the 10 MHz clock. The decoder starts operation after detecting the first rising edge of the received 4D symbol clock. After 23 clock cycles, the decoder stops and waits for the next rising edge of the 4D symbol clock. This allows the decoder to operate at any data rate from 0 to 2.1 Mbit/s.

Each iteration of the Viterbi algorithm decodes five bits for each received 4D signal point (since the code rate is 5/6). The maximum 4D symbol rate of the decoder is the internal clock speed divided the number of clock cycles required to decode the five bits, i.e., 4.35×10^5 4D symbols per second. Therefore, the maximum bit rate of the decoder is 2.17 Mbit/s. For the HST, this code could achieve a data rate up to 7.5 Mbit/s. For actual use on the HST, it is intended that the decoder would be implemented on a VLSI chip, where the required decoding speed would be achieved.

There are six main sections in the Viterbi decoder. These are

- Branch Metric Calculator (BMC)
- State Metric Calculator (SMC)
- Survivor Sequence Memory (SSM)
- Signal Set Synchronisor (SSS)
- Minimum State Metric Selector (MSMS)
- Branch Point Selector (BPS)

Figure 2 illustrates a block diagram of the decoder. The above sections are described as follows.

3.1 Branch Metric Calculator

For each transition of the trellis there are 8 parallel paths (due to the three unchecked bits in the encoder). The BMC must determine which of the paths is closest to the received 4D signal point (the Branch Point (BP)) as well as the Branch Metric (BM) for this path. The BM can be calculated in a number of ways. The optimum BM's for AWGN channels with quantisation are log-likelihood metrics [4]. Alternatively, one could make an approximation based on the squared Euclidean distance between the received point and the points along the transitions.

In our design we have chosen to use Read Only Memory's (ROM's) to store the precalculated BP (three bits are used to represent each parallel path) and BM (based on log-likelihood metrics). The encoder can produce one of eight (i.e., 2^{k+1}) sets of parallel paths (each containing 8 paths). The BP and BM must be determined for each of these eight sets of parallel paths.

We have chosen four bits to represent the BM value. This gives a BM range from 0 (closest to the received 4D point) to 15 (furthest from the 4D point). Decoder simulations in [5] for another multi-D trellis code indicate that this amount of quantisation results in little performance degradation.

To minimise the number of address bits to the ROM, each received 2D signal point has been quantised to seven bits. After extensive simulations in [5] for a 6D-8PSK trellis code, it was found that pie-chart or angular quantisation results in the least performance degradation (0.2 to 0.3 dB for five bit quantisation). The simulations included the "dartboard" quantisation pattern proposed in [1].

Each ROM therefore has an address space of 14 bits (seven bits for each 2D symbol). The ROM's used for the BMC are 32K×8 27C256's. A total of 6 ROM's were used, two for determining the BP's and four for the eight BM's.

Alternative BMC schemes which exploit the finite length trellis structure of the parallel transitions were also considered. That is, a Viterbi like decoder can be used to decode the parallel transitions. However, their large complexity (in a discrete implementation) led us to choose the simpler ROM look-up method. For a VLSI implementation, though, the trellis decoding method would be preferable due to the flexibility that VLSI provides in designing circuits. Thus, the Viterbi decoder (with the BMC) could be implemented on a single chip.

3.2 State Metric Calculator

The SMC updates the State Metrics (SM) for each state of the code in each iteration of the Viterbi algorithm. A SM is an indication of how close the received sequence is to the closest path of all paths leading into a particular state. Since the code has two checked bits, there are four paths leading into each state (since we choose the closest path among the 8 parallel paths in the BMC). For each of the four paths, we must add the BM for that path to its corresponding SM (also known as the old SM) from the previous iteration. The new SM for the four paths leading into a state is the smallest of these summations. This path is selected and all other paths are eliminated. This is called the Add-Compare-Select (ACS) operation.

With four paths into each state a 4:1 ACS circuit is required. With 16 states in our code, the ACS operation needs to be performed 16 times (explaining the need for 16 clock cycles). The ACS circuit also produces two Path Decision (PD) bits which indicate which of the four paths was chosen. This information is passed to the SSM where it is stored.

Since the decoder operates serially, only one ACS circuit is required. The 16 SM's are stored in two 74AS870 dual 16×4 static Random Access Memory (RAM) chips. Eight bits are used to represent each SM. As shown in [5] for a 6D-8PSK trellis code, this is more than enough bits when two's complement arithmetic is used in the ACS circuit to prevent overflow [4]. Before the first new SM can be calculated, four old SM's are read out from the RAM's. This takes four clock cycles. It takes another two clock cycles to perform the ACS operation. To achieve a slightly higher speed, we could have done the ACS operation in one clock cycle. However, this would have required six comparator chips to find the minimum SM. An increase of one clock cycle and the use of three comparator chips was chosen to decrease the complexity of the design.

Another clock cycle is used to write to the other half of the dual 16×4 RAM's. Since all the read and ACS operations are pipelined, an additional 15 clock cycles are required to write the 15 remaining new SM's. In the next iteration of the algorithm we read from where the SM's were written in the previous iteration and write to where the old SM's had been stored. The process then repeats.

For the ACS circuit, the appropriate BM's must be added to the correct old SM's. Twelve quad 2:1 multiplexer chips and a copy of the convolutional encoder are needed to accomplish this task.

3.3 Survivor Sequence Memory

The SSM has two tasks. It must store the Path Decisions (PD's) generated by the SMC and "traceback" through the previously stored PD's to determine the final decoded bits for x^2 and x^1. This requires alternating write and read (for the traceback) operations on the memory. The traceback depth is the required number of PD sets (each set consists of 16 two bit PD's) that the SSM must trace back through.

The PD's must be stored in the remaining 16 clock cycles that are available. There are two ways this can be achieved. Storing two PD bits in each clock cycle or storing four PD bits in every other cycle, leaving the alternate cycle to perform part of the traceback. With the first method at least two separate memories are required since the traceback operation cannot be performed simultaneously with the storage of the new set of PD's (due to the design of memory chips). Since there is a finite amount of memory, the oldest PD set must be written over.

There is usually a point where one method is better than the other (in terms of the total memory size required) based on the number of clock cycles available and the traceback depth. A traceback depth of around 25 to 30 results in little performance degradation [5]. Comparing the implementation complexity of the two methods, the alternating read/write method proved superior.

With this design only eight clock cycles are available to perform a traceback. To maintain integer power of 2 address spaces for the memories (and thus efficiently use of practical memory designs), a traceback depth of seven is used for each SSM memory chip. To achieve the required traceback depth, four 64×4 memories are required. This gives a traceback depth of 28. The traceback is performed in a pipelined fashion, switching between memories when required and waiting for the next received set of data to continue with the traceback. Four separate memories are required since there are four tracebacks in operation at any one time.

Since there are no 64×4 RAM's commercially available, larger 256×4 93422A RAM's were used. This chip has separate input and output data buses which simplifies the SSM design. We use the state with the smallest SM to start the traceback. This is the best state the SSM could start with (since it corresponds to the path that is closest to the received signal) and helps give the decoder a slight performance improvement over choosing a random or a fixed state. The Minimum State Metric Selector (MSMS) provides the information needed to achieve this.

At the correct time and place in the circuit, the two decoded bits x^1 and x^2 are produced. The two bits are passed to the Branch Point Selector (BPS) where they are re-encoded to select one of the eight 3 bit branch points. The branch points are delayed by 34 4D symbol periods, 28 due to the traceback, 4 due to the pipeline delay in the traceback, and 2 due to the re-encoding of the decoded data.

The five decoded bits are then differentially decoded (optional) and then parallel to serial converted for the final decoder output. Precoding and postdecoding are optional as there are some communication systems that do not require phase synchronisation. For example, a burst modem can provide phase information in the preamble of a burst. A 74HC4046 PLL is used to generate the required five times clock for the serial data. This PLL is tuned to lock within 0 to 2 MHz, but as expected for PLL's the lower frequency limit will be somewhat greater than DC. The decoded data is also available in five bit bytes.

3.4 Signal Set Synchroniser

The SSS has the task of synchronising the decoder to the received sequence of 2D symbols. Since the signal set consists of two 2D signals, the decoder must synchronise to one of the two possible ways the received data can arrive.

The decoder is asynchronously locked to DATCLK, which is the received 2D symbol clock whose frequency has been divided by two. A delay of zero or one 2D symbol periods of DATCLK is used for timing synchronisation.

The SSS works by examining the rate of increase of the minimum SM from the MSMS. If the rate is high, this indicates that the decoder is out of synch and needs to be resynchronised. A variable threshold in the SSS is used for this purpose. If the threshold is exceeded, the SSS will toggle into the "arm symbol toggel" state.

If the threshold is again exceeded in the next V (V is a variable from 0 to 63) 4D symbol periods the decoder will toggle the 2D symbol delay (from zero to one or one to zero). The SSS then ignores the decoder for 128+V 6D symbol periods to allow the decoder to settle into its new signal set configuration.

If the threshold is not exceeded the SSS will "disarm" and return to its normal monitoring state.

4 OTHER DECODER FEATURES

The encoder and decoder are mounted within a 3U high 19 inch rack. On the front panel, two Light Emitting Diodes (LED's) are used to indicate the 2D symbol delay.

To test the decoder, the 2D symbol delay can be independently set to manual control. In this way, the SSS can be isolated from the rest of the circuitry so that any problems with the rest of the decoder can be fixed without the SSS interfering. It can also be used to test the SSS by manually introducing delays into the received signal. There are two switches used for this.

Two rotary type switches are used to select the format of the received data. One switch is used to select between 3 bit phase (corresponding to hard decision), 7 bit phase quantisation, 5 bit I and Q quantisation, or internal loopback mode. The other switch selects between signed magnitude, reverse binary, straight binary, or two's complement data formats for I and Q received data.

There are also switches for disabling the postdecoder from the decoder and the precoder from the encoder. The encoder has another switch to select between five bit parallel or bit serial data. The decoder also has a reset button to force all the SM's to zero. The encoder/decoder interface diagram is given in Figure 3.

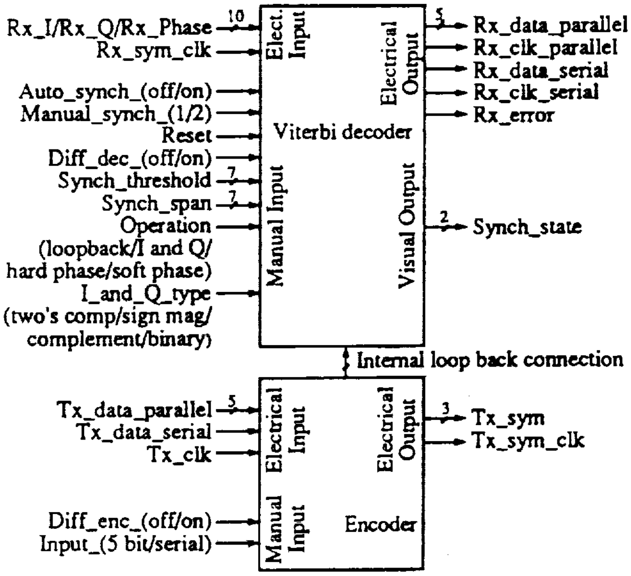


Figure 3: Viterbi decoder/encoder interface diagram for 16 state 2.5 bit/sym 4D-8PSK trellis code.

The 159 integrated circuits of the design are placed on two double height Speedwire Eurocards (233.4×220 mm). Speedwire allows quick and reliable connections (if it is done correctly) between the chips that can be easily changed. The speedwire boards also have good groundplanes, critical when operating at high clock speeds. The Viterbi decoder (which operates at 10 MHz) is placed on one board (taking 96 chips) while the encoder, SSS, and various interface chips are placed on the other board.

BNC connectors are used at the back of the rack for external data and clock connections. It is assumed that all received data changes on the rising edge of its clock. Similarly, the codec produces its signals in the same format. TTL 75 Ω interface signals are used for these external interfaces.

6 CONCLUSIONS

A serial implementation of a Viterbi decoder for the 16 state 2.5 bit/sym code with a 4D-8PSK signal set has been described. This decoder can provide high data rates (up to 2.1 Mbit/s) and is intended for future use on the Hubble Space Telescope. Due to its serial implementation the decoder design is quite complex, but could be implemented on a single VLSI integrated circuit.

The Branch Metric Calculator has been implemented through the use of large look-up table ROM's. A VLSI implementation may use a Viterbi type decoding algorithm to allow single chip implementation.

REFERENCES

[1] Pietrobon, S. S., R. H. Deng, A. Lafanechère, G. Ungerboeck, and D. J. Costello, Jr., "Trellis-coded multidimensional phase modulation," *IEEE Trans. Inform. Theory*, vol. 36, pp. 63-89, Jan. 1990.

[2] Perez, L., "On the performance of multi-dimensional phase modulated trellis codes," NASA Tech. Report #89-10-02, Oct. 1989.

[3] Porath, J. E., "Algorithms for converting convolutional codes from feedback to feedforward form and vice versa," *IEE Electron. Lett.*, vol.25, pp. 1008-1009, 20 Jul. 1989.

[4] Pietrobon, S. S., "Rotationally invariant convolutional codes for MPSK modulation and implementation of Viterbi decoders," M.Eng. Thesis, School of Electron. Eng., South Australian Inst. of Technol. (now University of South Australia), Adelaide, Australia, Jun. 1988.

[5] Gray, P. K., I. S. Morrison, and W. G. Cowley, "The development of a 45 Mbit/s modem/codec," *Proc. IREECON-'89*, pp. 942-945, Melbourne, Australia, Sep. 1989.

Appendix B

Construction of Trellis Codes With a Good Distance Profile

Construction of Trellis Codes with a Good Distance Profile [1]

Sanker S. Malladi, Fu-Quan Wang,
and Daniel J. Costello, Jr.

Department of Electrical Engineering
University of Notre Dame
Notre Dame, Indiana 46556

Hendrik C. Ferreira

Laboratory for Cybernetics
Rand Afrikaans University
P.O. Box 524, Johannesburg
2000 South Africa

Submitted to the
IEEE Transactions on Communications

July 1991

Abstract

Systematic feedforward trellis codes for 8-PSK and 16-QAM modulation are constructed using a nested step by step algorithm which guarantees a good distance profile. This makes the codes suitable for use with sequential decoding, where a rapidly growing distance profile is needed to reduce the average number of computations. In addition to having a good distance profile, the new codes achieve asymptotic coding gains up to 6.53 dB. A procedure based upon the Fano Algorithm (FA) is used to calculate the free distance of the new codes. This procedure is very effective for finding the free distances of long trellis codes because of the computational and storage efficiency of the FA. From a comparison of the new systematic feedforward codes with Ungerboeck's systematic feedback codes, we conjecture that a systematic feedforward code of constraint length 2ν can achieve the same free distance as a systematic feedback code of constraint length ν.

[1] This work was supported by NSF grant NCR 89-03429 and NASA grant NAG 5-557.

1 Introduction

A trellis code can be represented as a convolutional code with mapping by set partitioning [1-3]. Usually, to send k infomation bits/symbol, a 2^{k+1} point two-dimensional signal constellation is used. The incoming data is grouped as a k-bit block and fed into a rate $R = k/(k+1)$ convolutional encoder. The encoded (k+1) bits are then mapped to a point (or symbol) in the 2^{k+1} point signal constellation. Once the mapping is chosen, the performance of trellis codes is determined by the selection of the convolutional code. Thus, the construction of trellis codes involves selecting a convolutional code to optimize the minimum free Euclidean distance, the distance spectrum, and/or the distance profile depending on whether Viterbi decoding or sequential decoding is being used. In this paper, we construct codes with good distance profiles for 8-PSK and 16-QAM signal constellations for use with sequential decoding.

Convolutional encoders can be categorized as systematic feedforward, systematic feedback, and non-systematic feedforward. Only systematic feedback and non-systematic feedforward encoders are capable of generating optimum free distance codes. In general, there are many non-systematic feedforward encoders which can generate a given convolutional code. An encoder is minimal if it requires the fewest number of memory elements needed to generate a code [4]. In order to find a minimal encoder, it is always possible to convert a non-systematic feedforward encoder to an equivalent systematic feedback encoder [5]. The systematic feedback encoder is unique, minimal [4], and can never be catastrophic [6,7]. Also, rate $k/(k+1)$ encoders in systematic feedback form simplify computer searches for good trellis codes since there is only one parity check equation whose coefficients must be varied. Thus, most trellis codes are constructed in systematic feedback form. Ungerboeck [1], Porath and Aulin [8], and Pietrobon et. al. [9,10] have constructed systematic feedback trellis codes for a variety of signal constellations. Wei [11-14] has constructed both systematic feedback and non-systematic feedforward trellis codes. In this paper, since we are more concerned with the distance profile than the free distance or the distance spectrum, systematic feedforward codes are constructed.

The class of systematic feedforward codes cannot achieve the same performance (free distance) as systematic feedback or non-systematic feedforward codes with the same encoder memory (constraint length). However, this class of codes is capable of achieving a fast column distance growth (distance profile), which allows a sequential decoder to resynchronize rapidly [15,16]. Since the computational complexity of a sequential decoder is essentially independent of the code constraint length, longer codes can be used to achieve better performance (larger free distance). Thus, the class of systematic feedforward codes is a good choice for sequential decoding.

It has been shown by Chevillat and Costello [15,16] that a rapidly increasing Column Distance Function (CDF) results in a rapidly decreasing computational distribution and that the initial portion of the CDF affects the computational distribution of a sequential decoder more than the latter portion. The distance profile of a convolutional or trellis code is defined as its CDF over the first constraint length. Hence codes with good distance profiles will perform well with sequential decoding. Although the results of Chevillat and Costello were obtained for convolutuional codes, they are expected to hold for trellis codes also. Thus, in this paper, we construct trellis codes with good distance profiles for use with sequential decoding.

In Section 2, definitions of systematic feedforward codes and the column distance function for trellis codes are given. In Section 3, a nested step by step construction algorithm is used to find trellis codes with a good distance profile. A procedure based upon the Fano Algorithm (FA) is employed to evaluate the free distance of the codes constructed. In Section 4, the results are presented and a conjecture about the relationship between the free distance achievable with systematic feedforward and systematic feedback trellis codes is made. In Section 5, simulation results are presented to show that the new codes, which have better distance profiles than the Ungerboeck codes, result in a much better computational distribution and a better overall performance when used with sequential decoding. Finally, some conclusions are drawn in Section 6.

2 Notation and Definitions

A systematic feedforward trellis code can be generated by a rate $R = k/(k+1)$ systematic feedforward convolutional encoder along with mapping by set partitioning. The rate $R = k/(k+1)$ systematic feedforward convolutional code can be represented as

$$\mathbf{y} = \mathbf{x}\mathbf{G}, \tag{1}$$

where

$$\mathbf{y} = ({y_0}^k, {y_0}^{k-1}, \cdots, {y_0}^0; {y_1}^k, {y_1}^{k-1}, \cdots, {y_1}^0; \cdots\cdots) \tag{2}$$

$$\mathbf{x} = ({x_0}^k, {x_0}^{k-1}, \cdots, {x_0}^1; {x_1}^k, {x_1}^{k-1}, \cdots, {x_1}^1; \cdots\cdots) \tag{3}$$

are semi-infinite row vectors corresponding to the binary output and input sequences of the encoder, respectively,

$$\mathbf{G} = \left[\begin{array}{ccccccc} \mathbf{1}\mathbf{G}_0 & \mathbf{0}\mathbf{G}_1 & \cdots & \mathbf{0}\mathbf{G}_\nu & & & \\ & \mathbf{1}\mathbf{G}_0 & \mathbf{0}\mathbf{G}_1 & \cdots & \mathbf{0}\mathbf{G}_\nu & & \\ & & \mathbf{1}\mathbf{G}_0 & \mathbf{0}\mathbf{G}_1 & \cdots & \mathbf{0}\mathbf{G}_\nu & \\ & & & \ddots & \ddots & \cdots & \ddots \end{array} \right] \tag{4}$$

is a semi-infinite generator matrix, **1** is the $k \times k$ identity matrix, **0** is the $k \times k$ all-zero matrix, and

$$\mathbf{G}_i = \left[\begin{array}{c} g_i^k \\ g_i^{k-1} \\ \vdots \\ g_i^1 \end{array} \right] \tag{5}$$

for $i = 0, 1, 2, \cdots, \nu$, are the column generators which define the code. ν is the number of encoder memory elements and is called the constraint length of the code. For any constraint length ν code, $\mathbf{G}_\nu$ must not be zero.

A code can also be represented using polynomial notation. In this case, the binary output sequence $\mathbf{y}(D)$ is given by

$$\mathbf{y}(D) = \mathbf{x}(D)\mathbf{G}(D), \tag{6}$$

where

$$\mathbf{y}(D) = (y^k(D), \cdots, y^1(D), y^0(D)), \tag{7}$$

$$\mathbf{x}(D) = (x^k(D), \cdots, x^2(D), x^1(D)), \tag{8}$$

$$\mathbf{G}(D) = \begin{bmatrix} 1 & 0 & \cdots & 0 & G^k(D) \\ 0 & 1 & \cdots & 0 & G^{k-1}(D) \\ \vdots & \vdots & & \vdots & \\ 0 & 0 & \cdots & 1 & G^1(D) \end{bmatrix}, \tag{9}$$

and

$$G^j(D) = g_0^j + g_1^j D + \cdots + g_\nu^j D^\nu \tag{10}$$

for $j = 1, 2, \cdots, k$.

A general implementation of the systematic feedforward codes described above is shown in Figure 1. Note that some input information bits ($\tilde{k}+1$ to k) may be uncoded. In this case, the corresponding $G^j(D)$ ($j = \tilde{k}+1, \cdots, k$) are equal to zero. Encoders with some uncoded bits simplify code construction and decoding complexity, but limit the achievable free distance for larger constraint lengths. For short constraint lengths, however, encoders with some uncoded bits can give optimum free distance codes [1,9,10]. The uncoded bits introduce parallel transitions in the code trellis. For $\tilde{k} = 1$, parallel transitions limit the potential asymptotic coding gain to 3.0 dB, while for $\tilde{k} = 2$ and $\tilde{k} = 3$ the potential coding gains are limited to 6.0 dB and 9.0 dB, respectively. In this paper, all the information bits are coded for the 8-PSK ($k = \tilde{k} = 2$) and 16-QAM ($k = \tilde{k} = 3$) signal constellations considered in order to achieve the largest possible coding gains, since we aim to construct long codes which may achieve more than a 6.0 dB coding gain.

The signal mapper maps each binary encoder output (k+1)-tuple into one of 2^{k+1} possible signal points (symbols). This may be expressed as

$$a(\mathbf{y}_n) = (M_x(i_{\mathbf{y}_n}), M_y(i_{\mathbf{y}_n})), \tag{11}$$

where

$$\mathbf{y}_n = (y_n^k, \cdots, y_n^1, y_n^0) \tag{12}$$

is a binary encoder output vector at time unit n,

$$i_{\mathbf{y}_n} = 2^k \times y_n^k + \cdots + 2^1 \times y_n^1 + 2^0 \times y_n^0 \tag{13}$$

is the integer index which specifies the symbol $a(\mathbf{y}_n)$, and M_x and M_y are two functions which map the index of a symbol into its in-phase and quadrature values.

Definition 1. The Column Distance Function (CDF) of order i for a trellis code, d_i^2, is defined as

$$d_i^2 = \min_{\mathbf{x} \neq \mathbf{x}'} \left[\sum_{n=0}^{i} d^2 \left[a(\mathbf{y}_n), a(\mathbf{y}'_n) \right] \right], \tag{14}$$

where $\mathbf{x}$ and $\mathbf{x}'$ are two distinct information sequences and $d^2 \left[a(\mathbf{y}_n), a(\mathbf{y}'_n) \right]$ is the squared Euclidean distance between $a(\mathbf{y}_n)$ and $a(\mathbf{y}'_n)$.

Ungerboeck [1] defined the Euclidean weights

$$w^2(\mathbf{e}_n) \triangleq \min d^2[a(\mathbf{y}_n), a(\mathbf{y}_n \oplus \mathbf{e}_n)],$$

where $\mathbf{e}_n = [e_n^k, \cdots, e_n^1, e_n^0]$ is an n-bit error vector and the minimization is over all $\mathbf{y}_n = [y_n^k, \cdots, y_n^1, y_n^0]$, and he showed that there always exists a code sequence $(\mathbf{y}_0, \mathbf{y}_1, \cdots, \mathbf{y}_i)$ such that

$$\sum_{n=0}^{i} d^2 \left[a(\mathbf{y}_n), a(\mathbf{y}_n \oplus \mathbf{e}_n) \right] = \sum_{n=0}^{i} w^2(\mathbf{e}_n). \tag{15}$$

The significance of (15) is that the column distance function d_i^2 as well as the free distance d_{free}^2 of trellis codes can be calculated by assuming that the all zero sequence is sent. This simplifies the calculation of the free Euclidean distance of trellis codes in a way similar to the calculaltion of the free Hamming distance of convolutional codes. Thus, the Euclidean weights $w^2(\mathbf{e}_n)$ are used in the calculation of the column distance function d_i^2 as well as the free distance d_{free}^2.

Following [17], $\mathbf{d^2} = (d_0^2, d_1^2, \cdots, d_\nu^2)$ is called the distance profile of a trellis code. It has been shown [15,16] that the CDF should grow as rapidly as possible to achieve good performance for sequential decoding and that the initial part of CDF , i.e., the distance profile, plays a more important role than the latter part for sequential decoding.

Definition 2 [17]. A trellis code is said to have a distance profile $(d_0^2, d_1^2, \cdots, d_\nu^2)$ superior to the distance profile $(d_0'^2, d_1'^2, \cdots, d_\nu'^2)$ of another trellis code of the same constraint length ν if for some integer p, $0 \leq p \leq \nu$,

$$\begin{aligned} d_i^2 &= d_i'^2, \quad i = 0, 1, \cdots, p-1 \\ &> d_i'^2, \quad i = p. \end{aligned} \tag{16}$$

Since the class of systematic feedforward codes is suitable for use with sequential decoding and codes with good distance profiles perform well with sequential decoding, in this paper we have constructed systematic feedforward trellis codes with a good distance profile.

3 The Code Construction Algorithm

From (1)-(4) and (14), we see that $d_i^2(i \leq \nu)$ depends only on the column generators $(\mathbf{G_0}, \mathbf{G_1}, \cdots, \mathbf{G_i})$ for systematic feedforward codes. However, for large constraint lengths, it is impossible to conduct an exhaustive search to optimize the distance profile. Thus, we employed a nested step by step algorithm to construct systematic feedforward trellis codes with a good distance profile. This procedure is similar to Lin and Lyne [18], Costello [19], and Hagenauer's [20] methods for the construction of convolutional codes. The column generators $\mathbf{G_i}$ are selected as follows.

1) Choose the $\mathbf{G_0}$ which results in the maximum value of d_0^2, and set $i = 1$.

2) Suppose that $(\mathbf{G_0}, \mathbf{G_1}, \cdots, \mathbf{G_{i-1}})$ has been chosen. Choose the $\mathbf{G_i}$ that results in the maximum d_i^2. In case of a tie, the $\mathbf{G_i}$ that results in the minimum number of paths with distance d_i^2 is chosen.

3) If $i = \nu$, go to 4). Otherwise, set $i = i + 1$ and go to 2).

4) Evaluate the free distance of the code.

The above algorithm provides a simple construction for systematic feedforward codes because the distance profile up to the (i-1)-th stage does not change while searching for the best i-th column generator. Thus, previously chosen column generators do not have to be changed to maintain a good distance profile, i.e., the construction is nested. For systematic feedback codes, on the other hand, the coefficients of the parity check polynomials must be determined to specify a code. In this case, it is impossible (because of the feedback) to maintain the distance profile at the previous i-1 stages when selecting the coefficients of the i-th stage. Thus, the entire set of parity check coefficients must be changed at each stage of the algorithm to find systematic feedback codes with good distance profiles, i.e., the construction is not nested. This algorithm can also be used to construct non-systematic feedforward codes in a nested fashion. However, the construction of non-systematic feedforward codes is much more difficult since there exist many more generator coefficients at each stage.

To calculate d_i^2, all possible paths in the code trellis are extended at each step. The number of paths whose distances must be calculated increases exponentially with i. Thus, a distance cut-off value is introduced to simplify the algorithm. Only those paths whose distances do not exceed this cut-off value are stored for further extension. For systematic feedforward codes with natural mapping as used in this paper, it is easy to see that

$$d_{i+1}^2 - d_i^2 \leq \Delta_0^2, \tag{17}$$

where Δ_0 is the minimum distance between the signal points in a constellation. Without loss of generality, assume that the all zero sequence is sent. Then, (17) follows by noting that

$$\begin{aligned} {d_{i+1}}^2 &= d_i^2 + \min d^2[a(\mathbf{y}_{i+1}), a(\mathbf{y}'_{i+1})] \\ &= d_i^2 + \min w^2(\mathbf{e}_{i+1}) \\ &\leq d_i^2 + \Delta_0^2. \end{aligned} \tag{18}$$

The cut-off value can be estimated from the distance profiles of the shorter constraint length codes using (17).

To calculate the free distance d_{free}^2, a procedure based upon the Fano Algorithm (FA) is used. As pointed out in the last section, d_{free}^2 can be calculated by assuming that the all zero sequence is sent. The idea of using the FA to calculate the free distance is to decode a received sequence which is all zero. Thus, if the FA decoder is prevented from following any path starting with the zero symbol, it will find the non-zero path which has the best metric. By appropriately choosing the metric, the final metric of the decoded path will be the free distance of the code.

In the FA, if we set the metric to $-\infty$ if the first hypothesized symbol is zero, the decoder will never search a path starting with the zero symbol. This guarantees that a non-zero path will be decoded and that all non-zero paths will be explored. The metric is chosen to be 0 if the hypothesized symbol and a received symbol agree and $-d^2$ if they do not, where d^2 is the squared distance between the hypothesized symbol and the received symbol (this corresponds to the Euclidean weights $w^2(\mathbf{e}_n)$ defined in the last section). Since the path with the best metric is found by a FA decoder, the final decoded path will have the minimum distance to the all zero path, i.e., the final metric will be the free distance of the code.

Let Δ be the threshold increment and M_F be the cumulative metric of the decoded path. Initially, M_F, Δ, and the tentative free distance d_{free}^2 are set equal to a constant D known to be at least as large as the free distance of the code. This guarantees that Δ will not be lowered before the decoder finds a path that merges with the all zero path. In general, several decoding trials are needed before the free distance is found. At the beginning of each decoding trial, we set the initial path metric $M_F = \Delta$. At the end of each decoding trial, a new tentative free distance ${d_{free}^2}' = -M_F + \Delta$ is computed, where M_F is now the metric of the final decoded path. If ${d_{free}^2}' \geq d_{free}^2$, the previous tentative free distance, the algorithm stops. Otherwise, another decoding trial begins with a new (lower) tentative free distance $d_{free}^2 = {d_{free}^2}'$ and a lower $\Delta = d_{free}^2 - \delta$, where δ is chosen very small. Simulations show that the free distance can usually be found after only two or three trials. The free distance calculation procedure is described as follows.

1) Set $\Delta = D$, where D is some constant known to be larger than or equal to the free distance of the code. Set $d_{free}^2 = D$.

2) Set $M_F = \Delta$. Assume that the all zero sequence is received. Use the distance metrics defined above to replace the Fano metric used in the FA. Decode the received sequence until the decoder returns to the all zero state.

3) Calculate ${d_{free}^2}' = -M_F + \Delta$. If ${d_{free}^2}' \geq d_{free}^2$ go to 4). Otherwise, set $d_{free}^2 = {d_{free}^2}'$, $\Delta = d_{free}^2 - \delta$, where δ is a very small constant, and go to 2).

4) Print out d_{free}^2.

The selection of D is based upon known upper bounds on d_{free}^2. Several upper bounds on d_{free}^2 for trellis codes are available [21]–[23]. Although they are derived for codes in systematic feedback form, the bounds can be used as a good estimate of D. Actually, the procedure is not sensitive to the selection of D as long as D is larger than the free distance of the code, i.e., even starting with a very large D, the free distance can be found after only a few trials. This is because the tentative free distance found after the first trial is usually very close to the true free distance. Since systematic feedforward codes achieve smaller free distances than systematic feedback codes with the same constraint length, the above bounds are applicable here. δ should be chosen smaller than the gap between the free distance and the next smallest distance. However, since we do not know how small this gap is, a safe bet is to choose δ very small, say $\delta < 0.001$.

Rouanne and Costello [24] used a procedure based upon a bidirectional stack algorithm to calculate the entire distance spectrum of trellis codes. The procedure presented above is more efficient in calculating just the free distance of trellis codes. Forney [25] was the first one to suggest the use of sequential decoding to evaluate the distance spectrum of convolutional codes.

4 Results and Discussion

Table I and II show the resluts of a computer search for 8-PSK and 16-QAM trellis codes with a good distance profile, where the row generator $\mathbf{G}^j$ is defined as

$$\mathbf{G}^j = (g^j_\nu, g^j_{\nu-1}, \cdots, g^j_0), \;\; j = 1, 2, \cdots, k. \tag{19}$$

All the $\mathbf{G}^j$'s are expressed in octal form. (Note the difference between the row generators $\mathbf{G}^j$ and the column generators $\mathbf{G}_\mathrm{i}$.) The minimum squared column distance d^2_ν, the minimum squared free distance d^2_{free}, and the asymptotic coding gains are also listed in the tables. Δ_1 is the minimum distance between the points in a corresponding uncoded 2^k point constellation. The asymptotic coding gain γ of each code compared to the uncoded case is given by

$$\gamma = 10 \log_{10}(d^2_{free}/\Delta^2_1) \; dB. \tag{20}$$

Note that some codes in the tables have identical d^2_ν, d^2_{free}, and γ as for the previous (shorter) constraint length. This does not mean that the longer codes perform the same as the shorter ones. Simulations show that the longer codes usually perform better. This may be attributed to the fact that the longer codes have a smaller number of nearest neighbors.

Cedervall and Johannesson [26] have noted that systematic rate 1/2 convolutional codes of constraint length 2ν have about the same free distance as non-systematic codes of constraint length ν, confirming an old conjecture by Massey. After a careful comparison of our systematic feedforward codes with Ungerboeck's systematic feedback codes (equivalent to non-systematic feedforward codes), we conjecture that systematic feedforward codes of constraint length 2ν have about the same free distance as systemactic feedback codes of constraint length ν for trellis coded 8-PSK and 16-QAM. For example, the systematic feedforward codes of constraint lengths 9 and 18 in Table I have free distances of $d^2_{free}/\Delta^2_1 = 2.46$ and 3.46, respectively. On the other hand, the systematic feedback codes of constraint lengths 4 and 8 from [1] have free distances of $d^2_{free}/\Delta^2_1 = 2.58$ and 3.46, respectively. This indicates that about twice the constraint length is required for systematic feedforward codes to achieve the same free distance as systematic feedback codes. (It should be noted that the codes constructed in this paper may not have optimum free distance.) For trellis coded 16-QAM, we note that the systematic feedforward codes of constraint lengths 6 and 12 in Table II have the same free distances as the systematic feedback codes of constraint lengths 3 and 6 from [1], namely, $d^2_{free}/\Delta^2_1 = 2.5$ and 3.5, respectively.

Our only objective in the construction of these codes was to achieve a good distance profile, which is important for sequential decoding. It is noted that the selection of $\mathbf{G}_i$ in the construction algorithm is based upon the resulting column distance d_i^2. When several $\mathbf{G}_i$'s result in the same d_i^2, the one that gives the minimum number of paths with distance d_i^2 is selected. However, we cannot be sure whether some other $\mathbf{G}_i$ that results in the same d_i^2 at this stage will result in a larger d_j^2 at a later stage ($j > i$). Thus, the construction algorithm does not guarantee that the codes found have an Optimum Distance Profile (ODP), but an exhaustive construction of short codes did not find any codes with a better distance profile. From the steps used in selecting $\mathbf{G}_\mathrm{i}$, and noting that d_i^2 depends only on $(\mathbf{G}_0, \mathbf{G}_1, \cdots, \mathbf{G}_\mathrm{i})$, we are quite confident that the codes constructed using this algorithm are very close to ODP codes.

5 Simulation Results

Chevillat and Costello [15,16] have shown by analysis and simulations of convolutional codes that codes with better distance profiles outperform other codes when used with sequential decoding, both in the average number of compuations and in the distribution of computational effort. This motivated us to construct the trellis codes with good distance profiles listed above. Since, as pointed out in Section 1, once a mapping is chosen, the performance of a trellis code is determined by the selection of the convolutional code, we may expect that the observation of Chevillat and Costello with regard to the influence of the distance profile on decoding speed for convolutional codes will also hold for trellis codes. In this section, simulation results are presented to verify this.

We simulated a Fano sequential decoder for rate 2/3 trellis coded 8-PSK and decoded noisy data generated from a two-dimensional Gaussian noise distribution. Each simulation involved generating one thousand random sequences of 0's and 1's, encoding each sequence using rate a 2/3 binary convolutional code, and mapping the encoder outputs into frames (a fixed number of encoded symbols) of 8-PSK signal points. Each frame consisted of a sequence of 128 8-PSK symbols, and a sequence of 128 two-dimensional Gaussian noise vectors was added to form the received sequence. The received symbols were quantized into 8-PSK signal points using hard decisions. Then, the quantized received sequence was decoded by a simulated Fano sequential decoder. Simulations were performed on a SUN 3/50 computer.

In Figure 2, we show the distance profiles of two different rate 2/3, constraint length $\nu = 9$, 8-PSK trellis codes. The UG code was constructed by Ungerboeck [1]. The GDP (Good Distance Profile) code was taken from Table I. The GDP code clearly has a faster growing column distance function than the UG code. In Figure 3, we plot the computational distribution $Pr(C > N)$ of both codes for the Fano sequential decoder described above. $Pr(C > N)$ is defined as

$$Pr(C > N) = \frac{N_C}{N_F}, \tag{21}$$

where N_C is the number of frames for which the number of computations exceeded N and N_F is the total number of frames decoded. Each forward look was counted as one computation, and the simualtion was run at a signal-to-noise ratio (SNR) of $E_s/N_0 = 8.0$ dB, where E_s is the signal energy per transmitted symbol and $N_0/2$ is the noise power per dimension. It is clear that the computational distribution of the GDP code falls much more rapidly than the UG code. This indicates that trellis codes with good distance profiles will perform well with sequential decoding.

In Figures 4 and 5, we show the error performance for sequential decoding of the Ungerboeck (UG) code with constraint length $\nu = 8$ and the GDP code with $\nu = 18$. The UG code and the GDP code have the same free distance $d^2_{free} = 6.92$. An erasurefree version of the FA, called the Buffer Looking Algorithm (BLA) [27], was used for sequential decoding. In the BLA, an input buffer is used as in any other sequential decoding algorithm. The buffer is divided into two sections. When the first section of the buffer becomes full, suboptimum decoding is employed to force the decoder to finish decoding the current frame before the buffer overflows. In Figure 4, a buffer size of 4 K symbols, a decoder speed factor of 4, and a frame length of 256 symbols (512 information bits) were used. (The speed factor is defined as the number of computations that the decoder can perform during the time required to receive one symbol.) Figure 4 shows that the GDP code, which has the same free distance but a superior distance profile, performs better than the UG code.

Note that the performance curves in Figure 4 become closer at high SNR. Actually, the asymptotic performance of the two codes is expected to be the same since they have the same free distance. On the other hand, the average number of computations for sequential decoding decreases with increasing SNR. The difference in computational effort between the two codes will disappear eventually by noting that the average number of computations for both codes will approach one as the SNR approaches infinity. Thus, the performace curves of the two codes are expected to merge at some SNR for which the speed factor is much larger than the average number of computations. However, the merging SNR will be greater when a smaller buffer and/or a smaller speed factor are used. For example, Figure 5 shows the performance of the same two codes but with a (smaller) buffer size of 2 K symbols, a (smaller) decoder speed factor of 3, and a frame length of 256 symbols (same as above). Figure 5 shows that the GDP code performs better than the UG code just as in Figure 4, but that the merging SNR is about 0.5 dB greater.

The above simulation results show that the GDP codes constructed in this paper will perform better than the UG codes even though they have the identical free distance when sequential decoding is used. Also, even longer GDP codes, which have larger free distances, can be used to obtain even better performance with sequential decoding, since the computational complexity of a sequential decoder is essentially independent of the code constraint length.

6 Conclusions

In this paper, a step by step algorithm was utilized to construct trellis codes with good distance profiles. Systematic feedforward trellis codes for 8-PSK and 16-QAM modulation, with constraint lengths up to 25 and 15, respectively, were constructed. These new codes achieve asymptotic coding gains up to 6.53 dB. Simulations with sequential decoding show that trellis codes with better distance profiles outperform other codes in terms of computational effort and bit error rate. This is consistant with results previously found for convolutional codes. The trellis codes reported here are therefore recommended for use with sequential decoding.

A procedure based upon the Fano Algorithm (FA) was used to calculate the free distance of the new codes. This procedure is very effective for finding the free distance of long trellis codes because of the computational and storage efficiency of the FA. Comparing the new systematic feedforward codes with Ungerboeck's systematic feedback codes, we found that a systematic feedforward code of constraint length 2ν will have roughly the same free distance as a systematic feedback code of constraint length ν. Thus, we conjecture that systematic feedforward codes of constraint length 2ν will perform about as well as systematic feedback codes of constraint length ν.

References

[1] G. Ungerboeck, "Channel Coding with Multilevel/Phase Signals", *IEEE Trans. Inform. Theory*, **IT–28**, pp. 55–67, January 1982.

[2] G. Ungerboeck, "Trellis Coded Modulation with Redundant Signal Sets, Part I: Introduction", *IEEE Commun. Mag.*, **25**, pp. 5–11, February 1987.

[3] G. Ungerboeck, "Trellis Coded Modulation with Redundant Signal Sets, Part II: State of the Art", *IEEE Commun. Mag.*, **25**, pp. 12–22, February 1987.

[4] G. D. Forney, Jr., " Convolutional Codes I: Algebraic Structure," *IEEE Trans. Inform. Theory*, **IT–16**, pp. 720–738, November 1970.

[5] J. Porath, "Algorithms for Converting Convolutional Codes from Feedback to Feedforward Form and Vice Versa," *IEE Electron. Lett.*, **25**, pp. 1008–1009, July 1989.

[6] J. L. Massey and M. K. Sain, "Inverses of Linear Sequential Circuits," *IEEE Trans. Comput.*, **C–17**, pp. 330–337, April 1968.

[7] S. Lin and D. J. Costello, Jr., *Error Control Coding: Fundamentals and Applications*, Prentice Hall, New Jersey, 1983.

[8] J. Porath and T. Aulin, " Algorithmic Construction of Trellis Codes," submitted to the *IEEE Trans. Commun.*, November 1990.

[9] S. S. Pietrobon, R. H. Deng, A. Lafanechere, G. Ungerboeck, and D. J. Costello, Jr., " Trellis-Coded Multidimensional Phase Modulation," *IEEE Trans. Inform. Theory*, **IT–36**, pp. 63–89, January 1990.

[10] S. S. Pietrobon and D. J. Costello, Jr., " Trellis Coding with Multidimensional QAM Signal Sets," submitted to the *IEEE Trans. Inform. Theory*, April 1991.

[11] L. F. Wei, "Rotationally Invariant Convolutional Channel Coding with Expanded Signal Space-Part I: 180 degrees", *IEEE J. Select. Areas Commun.*, **SAC–2**, pp. 659–672, September 1984.

[12] L. F. Wei, "Rotationally Invariant Convolutional Channel Coding with Expanded Signal Space-Part II: Nonlinear Codes", *IEEE J. Select. Areas Commun.*, **SAC–2**, pp. 672–686, September 1984.

[13] L. F. Wei, "Trellis-Coded Modulation with Multidimensional Constellations", *IEEE Trans. Inform. Theory*, IT-33, pp. 483-501, July 1987.

[14] L. F. Wei, "Rotationally Invariant Trellis-Coded Modulations with Multidimensional M-PSK", *IEEE J. Select. Areas Commun.*, SAC-7, pp. 1281-1295, December 1989.

[15] P. R. Chevillat and D. J. Costello, Jr., "Distance and Computation in Sequential Decoding," *IEEE Trans. Commun.*, **COM–24**, pp. 440–447, April 1976.

[16] P. R. Chevillat and D. J. Costello, Jr., "An Analysis of Sequential Decoding for Specific Time-Invariant Convolutional Codes," *IEEE Trans. Inform. Theory*, **IT–24**, pp. 443–451, July 1978.

[17] R. Johannesson, "Robustly-Optimal Rate One-Half Binary Convolutional Codes," *IEEE Trans. Inform. Theory*, **IT–21**, pp. 464–468, July 1975.

[18] S. Lin and H. Lyne, " Some Results on Binary Convolutional Code Generators," *IEEE Trans. Inform. Theory*, **IT–13**, pp. 134–139, January 1967.

[19] D. J. Costello, Jr., "A Construction Technique for Random-Error-Correcting Convolutional Codes," *IEEE Trans. Inform. Theory*, **IT–15**, pp.631–636, September 1969.

[20] J. Hagenauer, "High Rate Convolutional Codes with Good Distance Profiles," *IEEE Trans. Inform. Theory*, **IT–23**, pp.615–618 , September 1977.

[21] A. R. Calderbank, J. E. Mazo, and H. M. Shapiro, "Upper Bounds on the Minimum Distance of Trellis Codes," *Bell Syst. Tech. J.*, **62**, pp. 2617–2646, October 1983, Part I.

[22] A. R. Calderbank, J. E. Mazo, and V. K. Wei "Asymptotic Upper Bounds on the Minimum Distance of Trellis Codes," *IEEE Trans. Commun.*, **COM–33**, pp. 305–310, April 1985.

[23] G. J. Pottie and D. P. Taylor, "Sphere-Packing Upper Bounds on the Free Distance of Trellis Codes," *IEEE Trans. Inform. Theory*, **IT–34**, pp.435–447 , May 1988.

[24] M. Rouanne and D. J. Costello, Jr., "An Algorithm for Computing the Distance Spectrum of Trellis Codes," *IEEE J. Select. Areas Commun.*, **SAC–7**, pp. 929–940, August 1989.

[25] G. D. Forney, Jr., "Use of a Sequential Decoder to Analyze Convolutional Code Structure," *IEEE Trans. Inform. Theory*, **IT–16**, pp. 793–795, November 1970.

[26] M. Cedervall and R. Johannesson, " A Fast Algorithm for Computing Distance Spectrum of Convolutional Codes," *IEEE Trans. Inform. Theory*, **IT–35**, pp. 1146–1159, November 1989.

[27] F. Q. Wang and D. J. Costello, Jr., "Erasurefree Sequential Decoding and Its Applications to Trellis Codes", presented at the 1990 IEEE International Symposium on Information Theory, San Diego, CA, January 1990.

Table I. Systematic Feedforward Trellis Codes for 8-PSK.

ν	$\mathbf{G}^1$	$\mathbf{G}^2$	$d_\nu{}^2/\Delta_1^2$	d^2_{free}/Δ_1^2	$\gamma(dB)$
3	12	0	1.59	1.59	2.00
4	32	20	1.59	1.59	2.00
5	52	0	1.88	1.88	2.74
6	152	100	1.88	1.88	2.74
7	252	0	2.00	2.00	3.00
8	652	400	2.17	2.17	3.37
9	652	1400	2.29	2.46	3.92
10	652	3400	2.46	2.46	3.92
11	652	7400	2.46	2.46	3.92
12	652	17400	2.46	2.46	3.92
13	20652	37400	2.76	2.76	4.41
14	60652	77400	2.76	2.76	4.41
15	60652	177400	2.76	3.05	4.84
16	260652	174400	2.76	3.17	5.01
17	660652	577400	2.76	3.34	5.24
18	1660652	577400	2.76	3.46	5.40
19	3660652	577400	2.87	3.46	5.40
20	7660652	577400	2.87	3.46	5.40
21	3660652	10577400	3.01	3.46	5.40
22	23660652	10577400	3.01	3.76	5.75
23	63660652	10577400	3.01	4.22	6.26
24	163660652	110577400	3.17	4.22	6.26
25	163660652	210577400	3.17	4.22	6.26

Table II. Systematic Feedforward Trellis Codes for 16-QAM.

ν	$\mathbf{G}^1$	$\mathbf{G}^2$	$\mathbf{G}^3$	$d_\nu{}^2/\Delta_1^2$	d_{free}^2/Δ_1^2	$\gamma(dB)$
3	16	04	04	1.5	1.5	1.76
4	16	24	24	2.0	2.5	3.98
5	56	64	64	2.0	2.5	3.98
6	156	164	164	2.0	2.5	3.98
7	256	064	264	2.5	2.5	3.98
8	656	464	364	2.5	3.0	4.77
9	1656	0464	1364	2.5	3.5	5.44
10	3656	2464	1364	2.5	3.5	5.44
11	7656	6464	5364	3.0	3.5	5.44
12	17656	16464	15364	3.0	3.5	5.44
13	37656	16464	15364	3.0	4.0	6.02
14	37656	56464	15364	3.0	4.5	6.53
15	137656	056464	015364	3.0	4.5	6.53

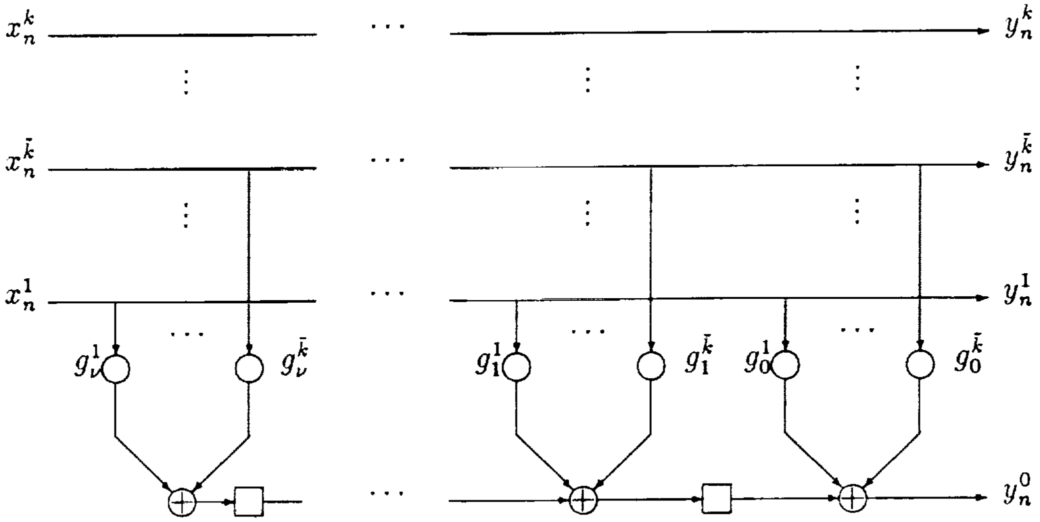


Figure 1. Implementation of a General Systematic Feedforward Convolutional Encoder.

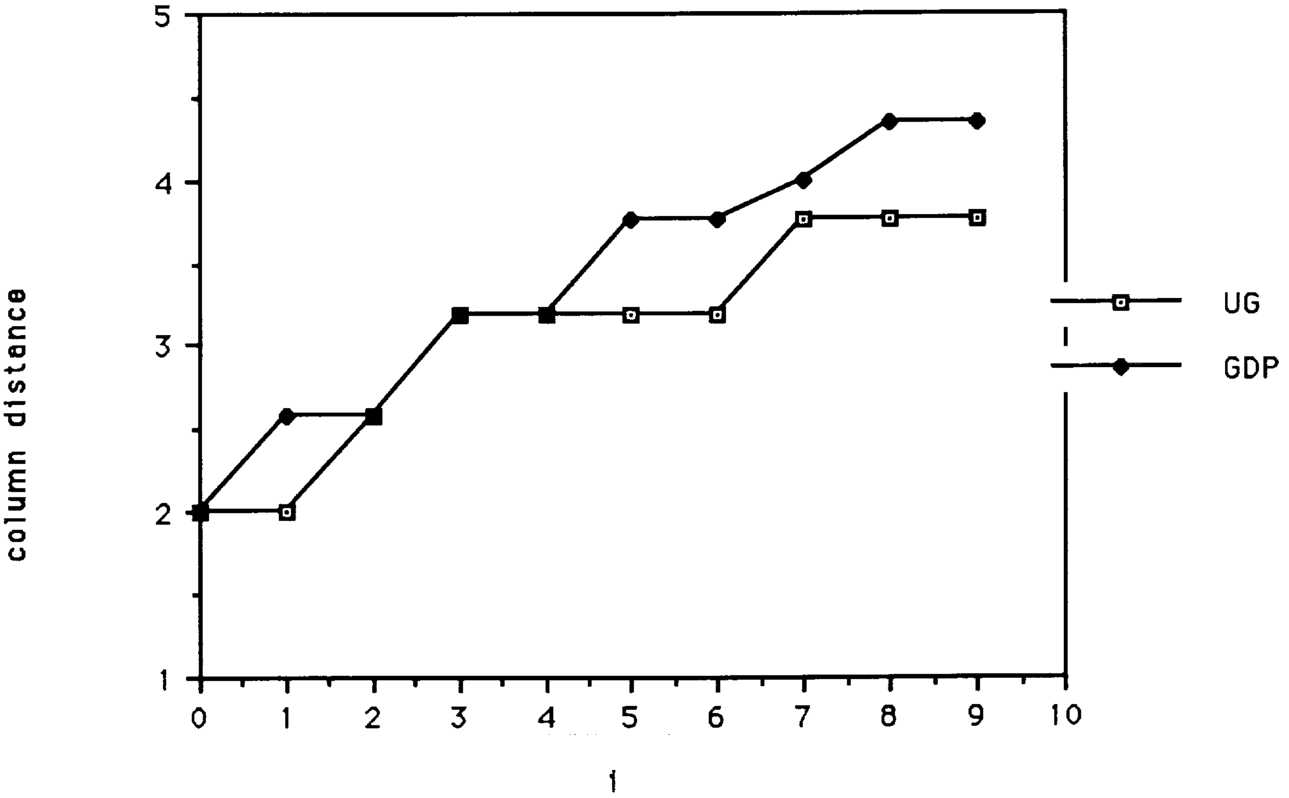


Figure 2. Distance profiles of two rate 2/3, constraint length 9 trellis codes

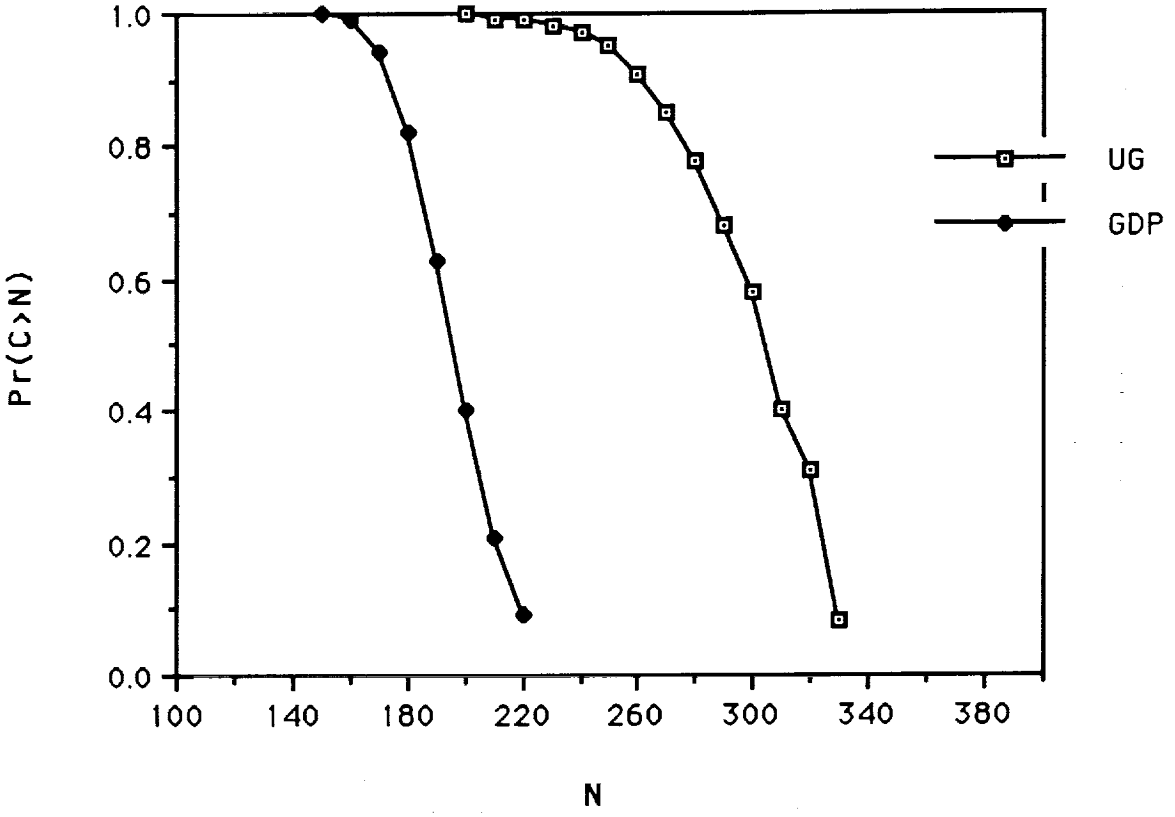


Figure 3. Computational distribution for two rate 2/3, constraint length 9 trellis codes at SNR=8 dB

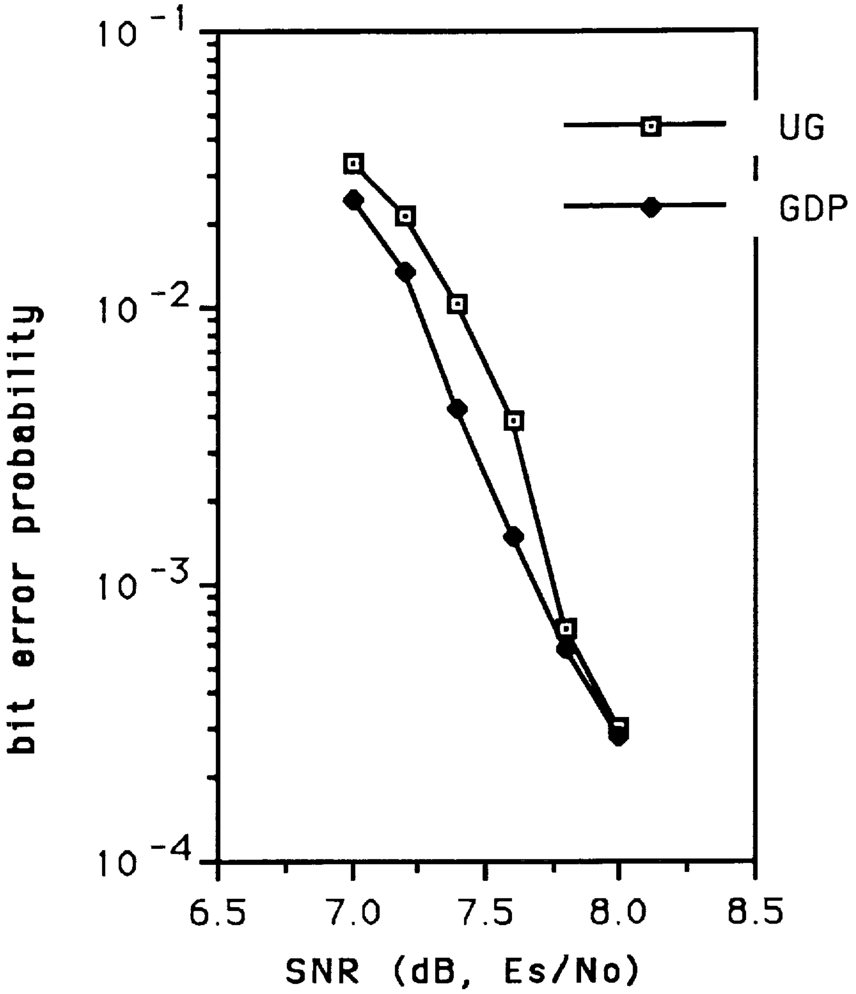


Figure 4. Performance of systematic feedforward and feedback codes

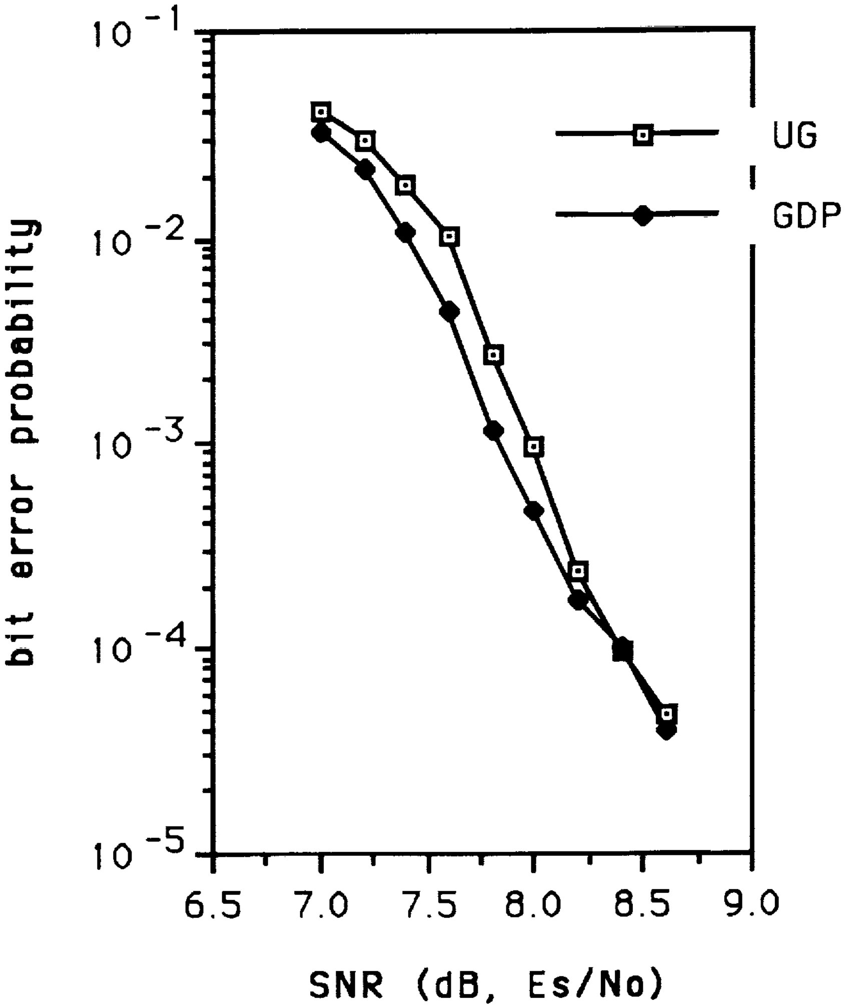


Figure 5. Performance of systematic feedforward and feedback codes

Appendix C
Probabilistic Construction of Trellis Codes

Probabilistic Construction of Trellis Codes*

Fu-Quan Wang
Daniel J. Costello, Jr.
Department of Electrical Engineering
University of Notre Dame
Notre Dame, Indiana 46556

presented at
the 1991 International Symposium
on Information Theory
Budapest, Hungary

June, 1991

*This work was supported by NSF Grant NCR 89–03429 and NASA grant NAG 5–557.

Outline of Presentation

- Preliminaries
- Performance of Short Codes with Viterbi Decoding
- Sequential Decoding vs. Viterbi Decoding
- Construction of Long Codes for Sequential Decoding
- Performance of Long Codes with Sequential Decoding
- Conclusions

Preliminaries

- There are two important parameters that determine fundamental performance limits for digital communication: the channel capacity C and the channel cut-off rate R_0.

- Channel capacity C is the maximum rate for which reliable communication can be achieved using coding. It can only be achieved with infinite coding complexity.

- Cut-off rate R_0 is the maximum rate at which the average number of computations for sequential decoding (SD) is bounded. R_0 is regarded as the maximum rate for which reliable communication can be achieved with reasonable complexity.

- We construct bandwidth efficient codes that can achieve the cut-off rate bound at bit error rates (BER's) of 10^{-5} to 10^{-6} using SD (the Fano algorithm is used throughout the paper).

Channel Model

- A discrete-input additive white Gaussian noise (AWGN) channel is assumed.

- Signal a_l is transmitted at modulation time lT. a_l is taken from a collection of signals $\{a^i, i = 0, 1, \cdots, K-1\}$ with probability Q(i) (i=0,1, $\cdots$, K-1). The average signal energy (per two dimensions) is:

$$S = \sum_{i=0}^{K-1} Q(i)||a^i||. \tag{1}$$

- Noise w_l, variance (per dimension) σ^2.

- Channel output:

$$z_l = a_l + w_l \tag{2}$$

with probability density:

$$p\{z_l/a_l = a^i\} = \frac{1}{2\pi\sigma^2} exp\{\frac{-|z_l - a^i|^2}{2\sigma^2}\}. \tag{3}$$

- Channel signal to noise ratio (SNR):

$$SNR = E_s/N_0 = S/2\sigma^2. \tag{4}$$

Channel Capacity and Cut-off Rate on the Gaussian Channel

- AWGN, no constraint on the input signal sets.

- Channel capacity (Shannon):

$$C = \log_2 \left(1 + \frac{E_s}{N_0} \right) . \tag{5}$$

- Cut-off rate (Shannon):

$$R_0 = (\log_2 e) \left[1 + \frac{E_s}{2N_0} - \sqrt{1 + \left(\frac{E_s}{2N_0} \right)^2} \right] + \log_2 \left[\frac{1}{2} \left(1 + \sqrt{1 + \left(\frac{E_s}{2N_0} \right)^2} \right) \right] . \tag{6}$$

- These expressions are independent of the signal sets, which are assumed to be optimum.

Channel Capacity and Cut-off Rate for Equiprobable Signaling on the Gaussian Channel

- Discrete-input, continuous-output channel.

- $Q(i) = \frac{1}{K}$, i=0,1, $\cdots$, K-1.

- Channel capacity (Ungerboeck):

$$C^* = \log_2 K - \frac{1}{K} \sum_{k=0}^{K-1} E_z \left\{ \log_2 \sum_{i=0}^{K-1} exp \left[- \frac{|z - a^i|^2 - |z - a^k|^2}{2\sigma^2} \right] \right\}. \tag{7}$$

- Cut-off rate (Wozencraft and Jacobs):

$$R_0^* = 2 \log_2 K - \log_2 \left\{ \sum_{i=0}^{K-1} \sum_{j=0}^{K-1} exp \left[- \frac{|a^i - a^j|^2}{8\sigma^2} \right] \right\}. \tag{8}$$

- C^* and R_0^* are computed for specific equiprobable signal sets and can be theoretically achieved by combining coding with those signal sets, while C and R_0 can only be approached with optimally shaped signal sets.

Shaping Gain

- QAM and PSK modulation are considered.

- For QAM modulation, a smaller SNR is required to achieve C than C^*, and a smaller SNR is required to achieve R_0 than R_0^*. The difference between these SNR's is the potential shaping gain that can be achieved with non-equiprobable signaling.

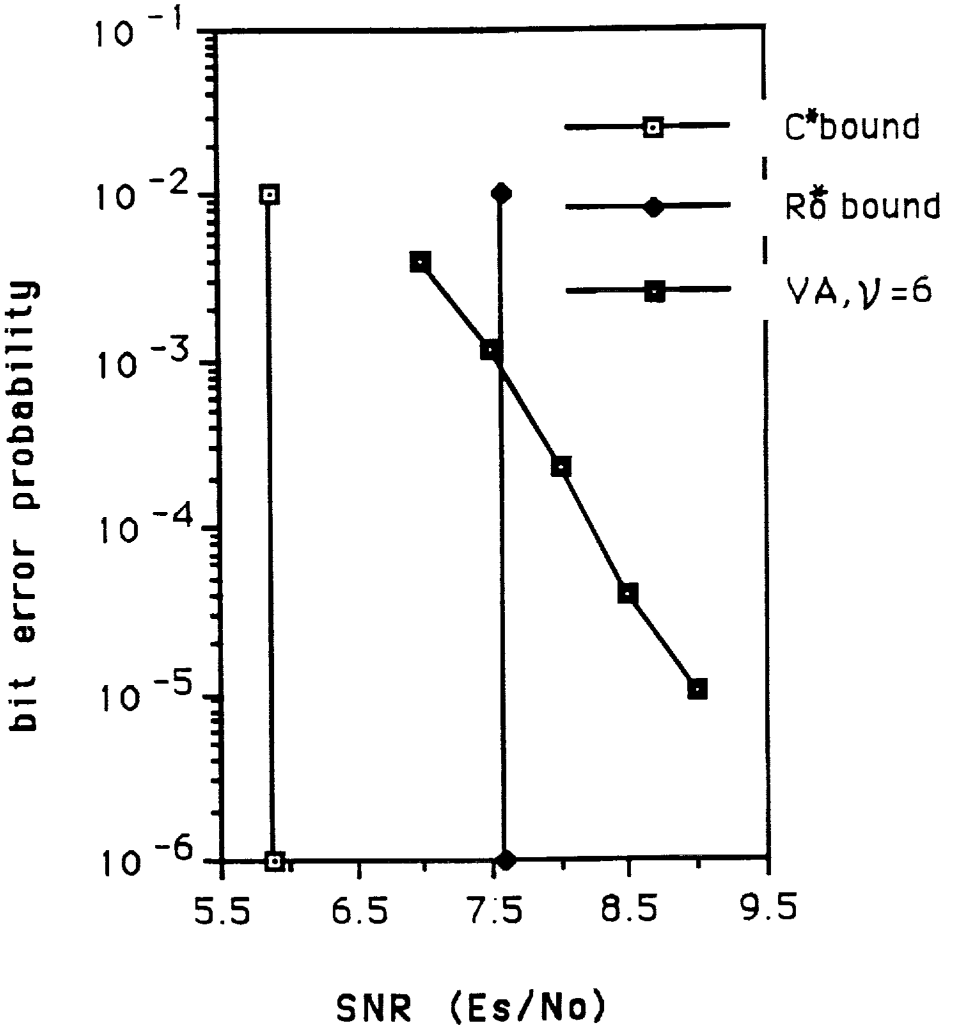
- No shaping gain exists for PSK modulation because the signals in the constellation all have the same energy.

- The maximum theoretical shaping gain is defined as the average energy saved by choosing signals in an N-dimensional sphere rather than an N-dimensional cube of the same volume as N approaches infinity. The maximum gain is about $\pi e/6$ or 1.53 dB.

- Usually, the potential shaping gain with respect to channel capacity is larger than the potential shaping gain with regard to cut-off rate. Thus, less shaping gain is available when SD is being used.

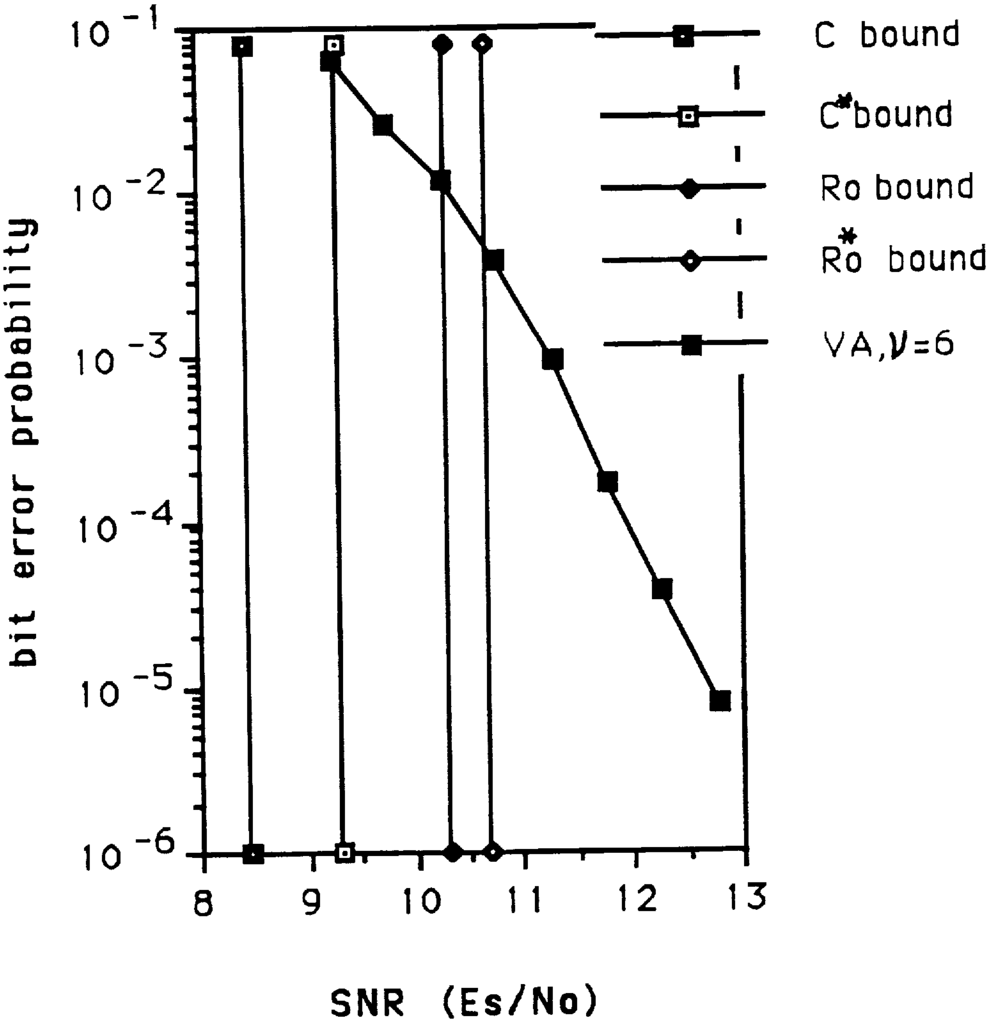
Trellis Coded 8-PSK Performance with Short Codes

- The performance of rate 2/3 trellis coded 8-PSK with constraint length $\nu = 6$ using the Viterbi algorithm (VA) is shown along with the channel capacity and cut-off rate bounds.
- At a BER of 10^{-5} the code is about 1.4 dB away from the R_0^* bound and 3.1 dB away from the C^* bound.
- The R_0^* bound can be approached using SD.



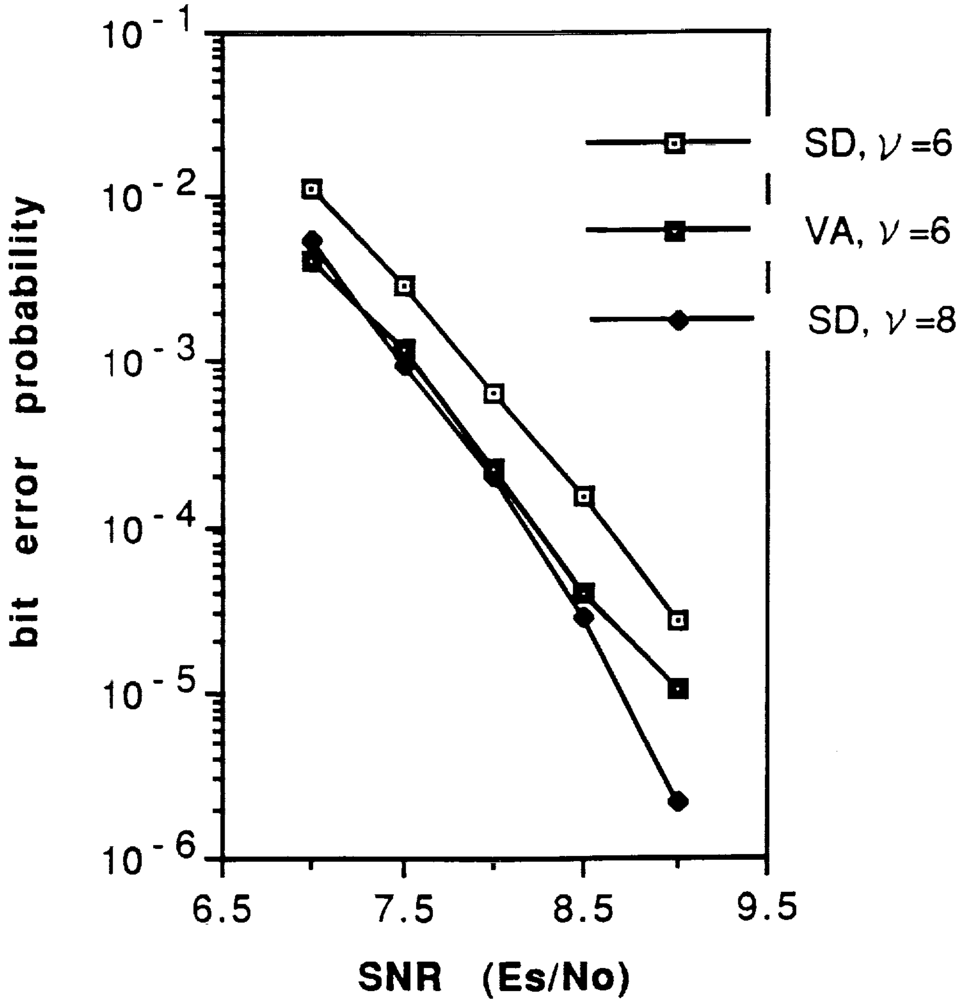
Trellis Coded 16-QAM Performance with Short Codes

- The performance of rate 3/4 trellis coded 16-QAM with constraint length $\nu = 6$ using the VA is shown along with the channel capacity and cut-off rate bounds.
- At a BER of 10^{-5} the code is about 2.0 dB away from the R_0^* bound and 3.4 dB away from the C^* bound.
- There is about 0.8 dB shaping gain available for 16-QAM with regard to channel capacity. There is only about 0.4 dB shaping gain available with regard to cut-off rate.
- The R_0 bound can be approached using SD with non-equiprobable signals and a shaping code.



Sequential Decoding vs. Viterbi Decoding

- The computational effort of the VA increases exponentially with ν while it is essentially independent of ν for SD.
- SD performs very close to maximum likelihood, but the decoding speed is variable and data is delivered asynchronously to the user.
- The performance of rate 2/3 trellis coded 8-PSK using Viterbi decoding and sequential decoding is shown below. Ungerboeck codes with $\nu = 6$ and $\nu = 8$ intended for use with the VA are used.
- Note that SD can overcome its suboptimum performance by using a slightly larger constraint length with no penalty in computational effort.



Code Construction Problem

- Criteria:
  - Free distance (for the VA)
  - Distance profile (for SD)
- Approaches:
  - Exhaustive search with rejection rules (Ungerboeck)
  - Hueristic construction (Wei)
  - Algorithmic construction (Porath and Aulin)
- Problems:
  - The error performance of a code is determined by its entire distance spectrum. Better free distance may not result in better performance.
  - Determination of free distance becomes very difficult for large constraint lengths.
  - For large constraint lengths, the set of codes becomes too large to conduct an exhaustive search.
- Codes will be constructed in systematic feedback form throughout this paper.

Results from Random Coding

- The average error probability of all rate $R = k/n$ trellis codes satisfies the bound (Viterbi and Omura)

$$P_{av}(e) < (2^k - 1)\frac{2^{-(\nu+1)kR_0/R}}{[1 - 2^{-\epsilon k R_0/R}]^2} \tag{9}$$

for $0 \leq R \leq R_0(1-\epsilon)$, ϵ is a positive constant, ν is the constraint length.

- Suppose there are a total of N codes and $P_i(e)$ is the error probability of the i-th code. Then,

$$P_{av}(e) = \frac{1}{N}\sum_{i=1}^{N} P_i(e) = \frac{1}{N}\sum_{i=1}^{K} P_i(e) + \frac{1}{N}\sum_{i=K+1}^{N} P_i(e). \tag{10}$$

Without loss of generality, suppose the first K (any positive integer less than N) codes have error probabilities greater than $\frac{N}{K}P_{av}(e)$. Then the first term in the above equation is larger than $P_{av}(e)$. Noting that $P_i(e)$ is always positive, this implies that fewer than K codes can have error probability greater than $\frac{N}{K}P_{av}(e)$, and hence at least $N-K$ codes must have error probability less than $\frac{N}{K}P_{av}(e)$.

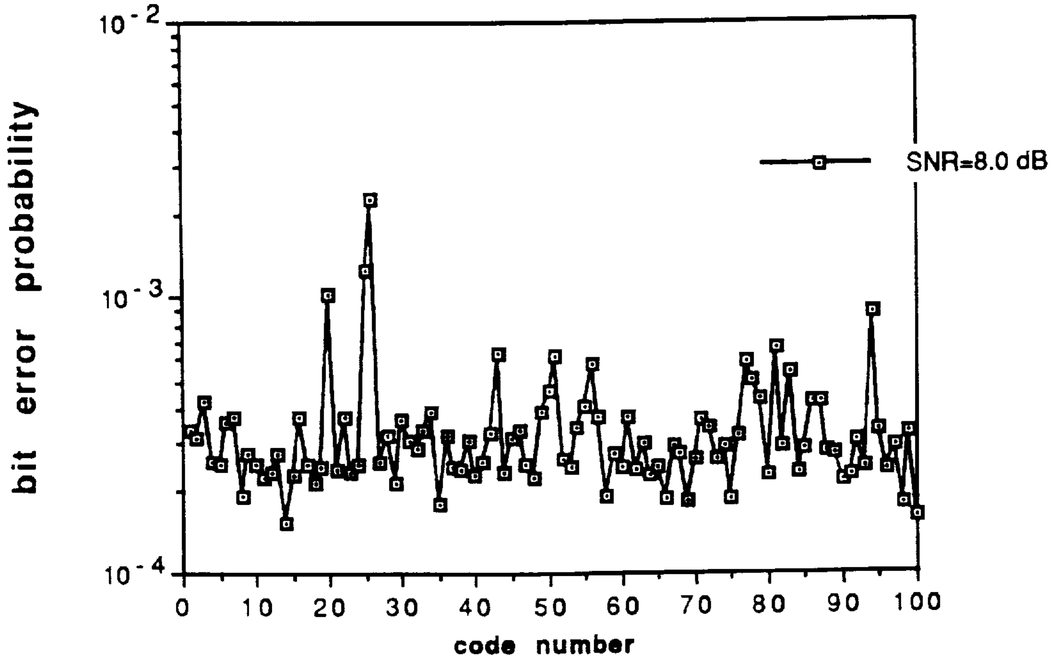
- Let $\lambda = \frac{K}{N}(0 < \lambda < 1)$. We conclude that at least a fraction $1-\lambda$ of all codes must have an error probability less than $\frac{1}{\lambda}P_{av}(e)$.

Performance of Randomly Chosen Codes

- Let $\lambda = 0.1$. Then at least 90% of all codes have error probability $P(e) < 10P_{av}(e)$.
  Let $\lambda = 0.5$. Then at least 50% of all codes have error probability $P(e) < 2P_{av}(e)$.

- The SD performance of a set of 100 randomly chosen rate 2/3 trellis codes for 8-PSK modulation with $\nu = 8$ and $SNR = 8.0dB$ is shown below.

- Note that several codes are found with very good performance.



An Approach to Constructing Good Long Codes

- The purpose of code construction is to determine the codes that give the best performance. Codes may be chosen based upon their distance properties or their actual performance.

- Noting that it may not be possible to evaluate the distance properties of long codes, the direct evaluation of performance may be the best practical way to construct long codes.

- The above discussion and simulations imply that some good trellis codes can be found from a randomly chosen set of codes.

- Since its computational effort is essentially independent of ν, SD can be used to determine the performance of a set of large constraint length codes chosen at random.

- Since a randomly chosen set of codes contains some good codes with high probability, it should be possible to find codes using this approach whose performance with sequential decoding meets the R_0 bound.

Another Approach: Simulated Annealing

- Simulated annealing is a computational heuristic for obtaining approximate solutions to combinatorial optimization problems.

- Code construction may be viewed as a combinatorial optimization problem where the parity check (or generator) coefficients are the variables and the free distance or the performance of a code is the objective (cost) function.

- Simulated annealing has been used to construct block codes (El Gamal and others).

- We try to construct good large constraint length trellis codes using simulated annealing. The SD performance is used as the cost function.

Construction Algorithm 1: Random Search

- Let N_c be the number of codes to be examined, N_b be the number of encoded sequences (each sequence consists of m information bits) to be decoded for each code, and P_b be the average bit error probability of a code.

- The basic algorithm:

1. Choose the SNR at which the codes are to be evaluated, N_c, and N_b. Let n_c and n_b be the number of codes examined and sequences decoded thus far, respectively. Set $n_c = 0$, $n_b = 0$, and $P_b = 1.0$.
2. Select a code by randomly choosing the generator (or parity-check) coefficients.
3. Encode a randomly chosen sequence of m information bits using the code chosen in 2.
4. Add channel noise to the encoded sequence.
5. Decode the corrupted sequence using sequential decoding. Set $n_b = n_b + 1$. If $n_b < N_b$, go to 3. Otherwise, go to 6.
6. Calculate the average bit error probability P_{bt} of the N_b encoded sequences. If $P_{bt} > P_b$, go to 8. If $P_{bt} \leq P_b$, go to 7.
7. Print P_{bt} and the generator (or parity-check) coefficients of the code. Set $P_b = P_{bt}$.
8. Set $n_c = n_c + 1$. If $n_c < N_c$, go to 2. Otherwise, stop.

- Some modifications can be made to speed up the construction.

- The information block size m is usually chosen to be 1000 bits.

Construction Algorithm 2: Simulated Annealing

- Let N_e be the number of energy drops required to lower the temperature, N_i be the number of iterations required to lower the temperature, and N_c be the number of consecutive temperature stages that produce no change in the code required to stop the code search.
- Define the energy (cost function) of a code C as $Energy(C) = P_b(C)$, where $P_b(C)$ is the average bit error probability of the code C at some SNR.
- The procedure:

1. Let n_e be the number of energy drops, n_i be the number of iterations, and n_c be the number of consecutive temperature stages that produce no change in the code. Choose a code C and a temperature T. Let $n_e = 0$, $n_i = 0$, and $n_c = 0$.

2. Choose a code C', a perturbation of C (randomly "jiggle" one coefficient). Let $\Delta E = Energy(C') - Energy(C)$. If $\Delta E < 0$, $C \leftarrow C'$ and $n_e = n_e+1$. Otherwise, with probability $\exp(-\Delta E/T)$, $C \leftarrow C'$. If $C \leftarrow C'$ occurs, let $n_c = 0$.

3. $n_i = n_i + 1$.

4. If $n_e \geq N_e$, go to 6. Otherwise, go to 5.

5. If $n_i \geq N_i$, go to 6. Otherwise, go to 2.

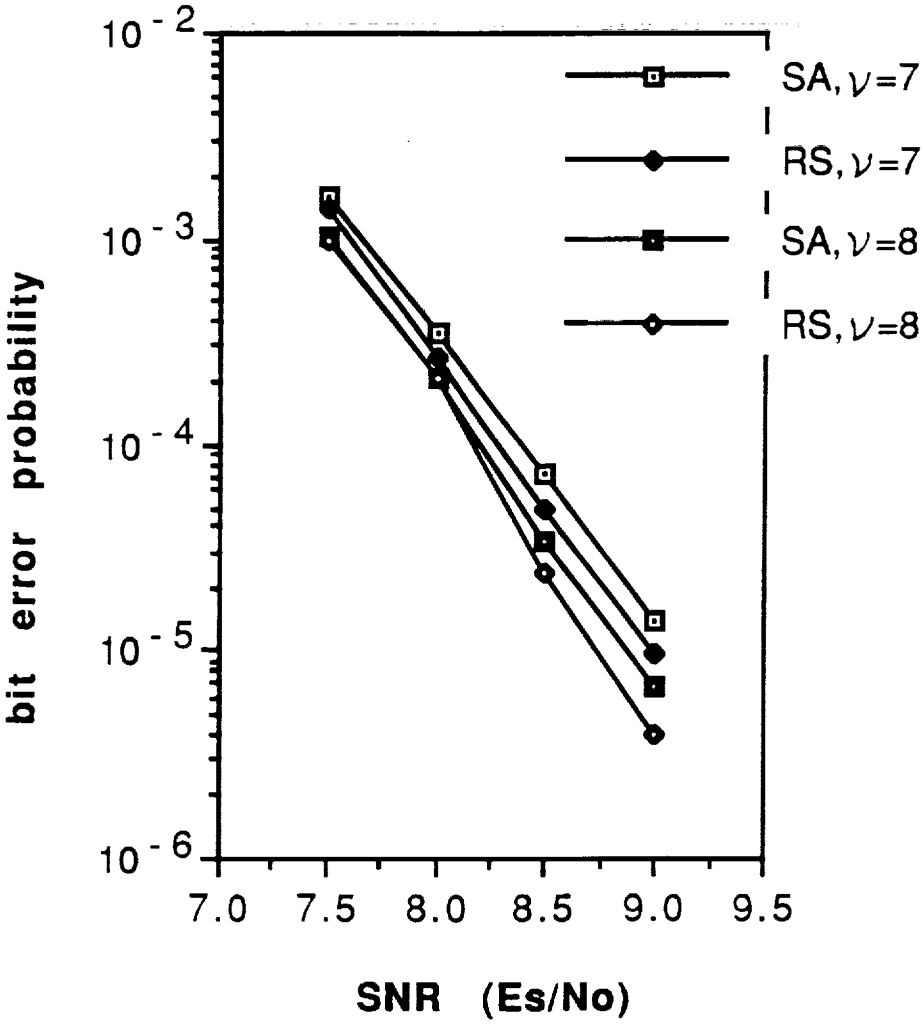
6. Let $n_e = 0, n_i = 0, n_c = n_c + 1$, and T $\leftarrow \alpha T$ $(1 > \alpha \geq 0.9$, a constant). If $n_c < N_c$, go to 2. Otherwise, print out the code generator (or parity-check) coefficients and stop.

- Typical threshold values are $N_e = 3$, $N_i = 20$, and $N_c = 5$. A code with all zero coefficients (a poor code) is chosen as the initial C. T is usually chosen to be roughly one hundred times the expected BER of the best code.

Comparison of the Two Algorithms

- Treilis codes for 8-PSK modulation with constraint lengths $\nu = 7$ and 8 are constructed. A total of 200 codes are evaluated using the random search while several hundred to several thousand codes are evaluated using simulated annealing. The codes are evaluated at an SNR=7.75 dB.

- The performance of the best codes constructed is compared below. It shows that the codes constructed by the two algorithms perform almost the same.



Comments on the Construction Algorithms

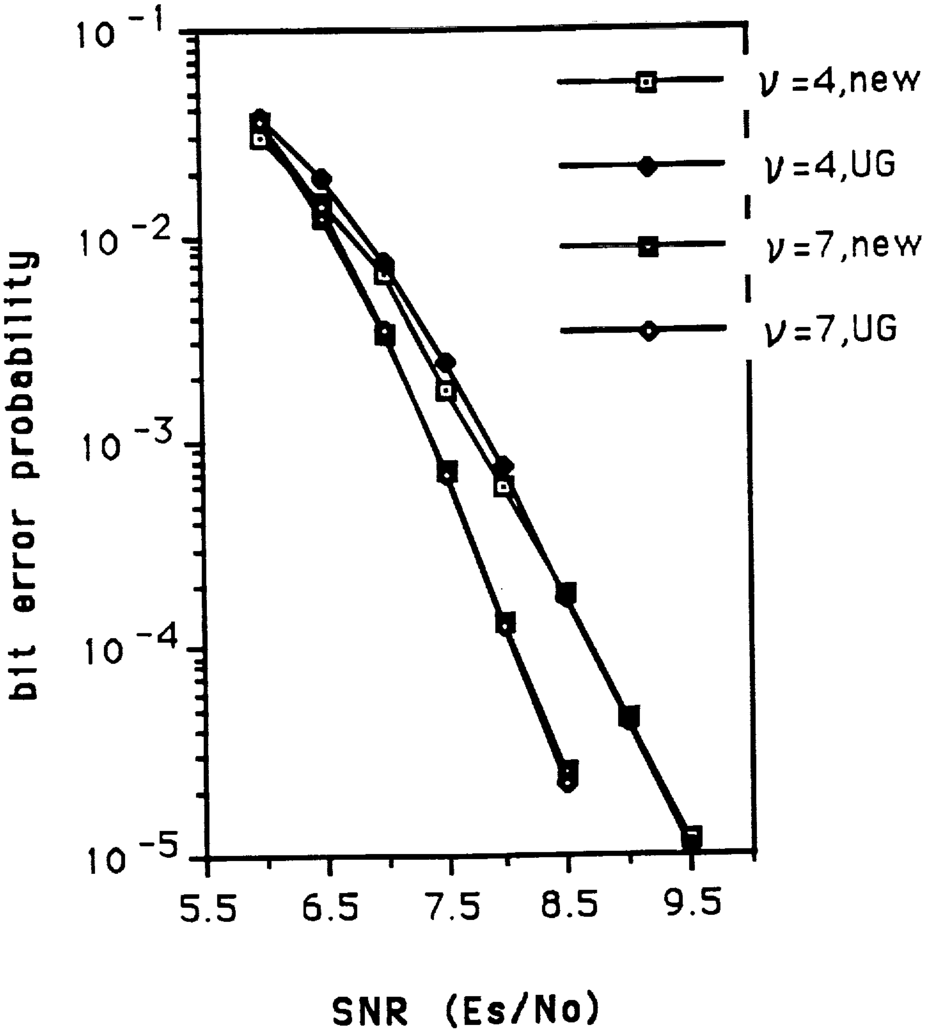
- Our confidence in the performance evaluation of a code depends on the number of errors decoded. Usually several hundred errors are decoded for each code evaluated.

- To insure that good codes are found, two steps are employed. First, several codes that perform well at the chosen SNR are obtained from the search procedure. These codes are then evaluated over a wide range of SNR's with much more data being decoded. This allows us to select the best code with a high degree of confidence.

- Although many more codes have been tested using simulated annealing, the codes obtained using this approach actually perform slightly worse than the codes found by the random search procedure. This may be attributed to the fact that simulated annealing tends to lead the search to a local minima.

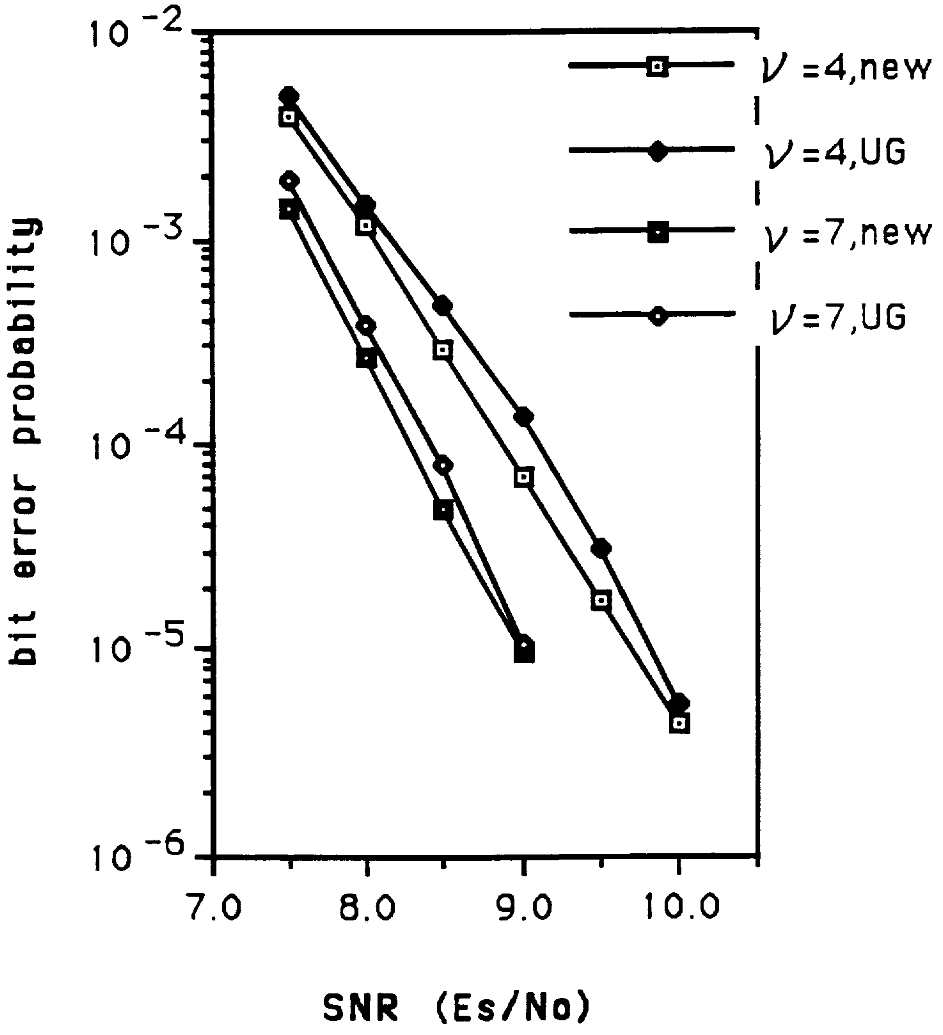
Comparison of New Short Codes with Ungerboeck Codes (Viterbi Decoding)

- Trellis codes for 8-PSK modulation with $\nu = 4$ and $\nu = 7$ are constructed using the random search algorithm with $N_c = 200$ and decoded using the VA. The performance of the new codes along with Ungerboeck codes of the same constraint length is shown below.
- At low SNR, the new codes perform slightly better than the Ungerboeck codes. This is due to the fact that the Ungerboeck codes have larger path multiplicities than the new codes.
- A calculation of the distance spectrum shows that in many cases the new codes have smaller multiplicities but less free distance than the Ungerboeck codes. This is because the codes are constructed at a low SNR.



Comparison of New Short Codes with Ungerboeck Codes (Sequential Decoding)

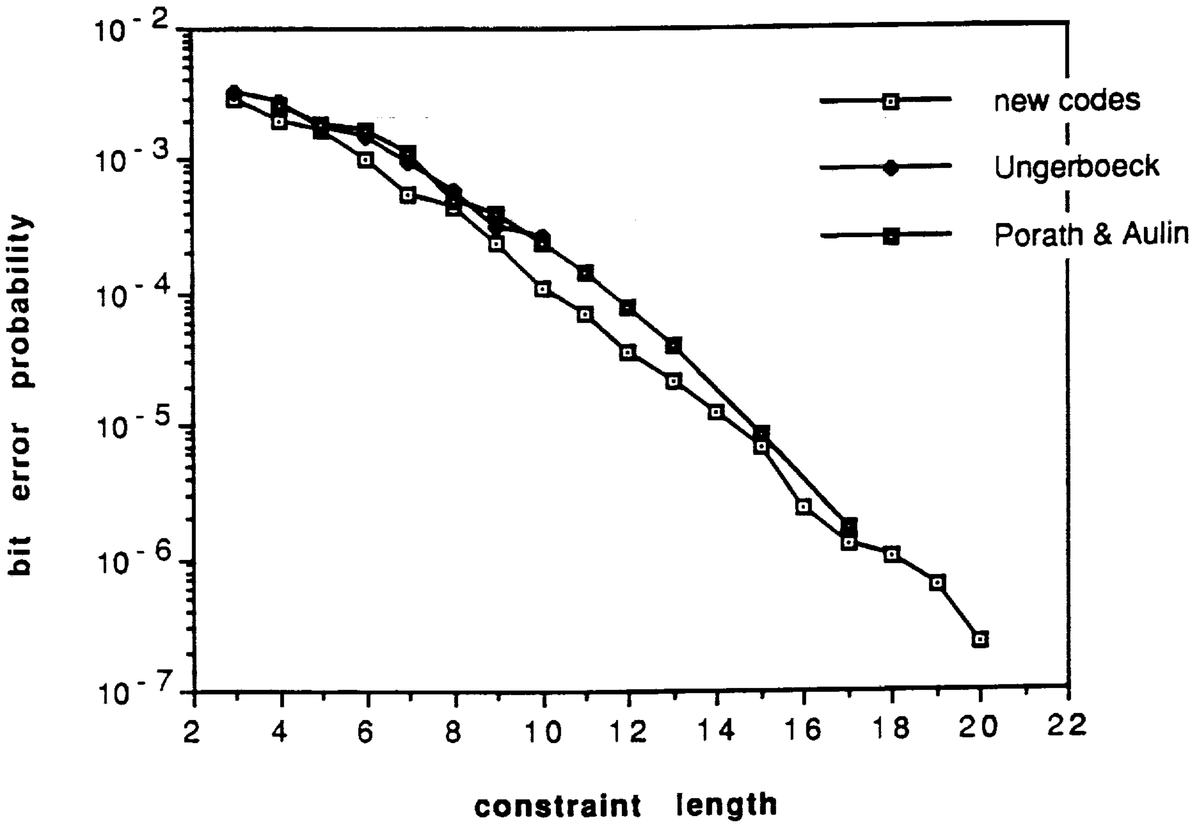
- The same codes are now decoded using SD. Their performance is shown below.
- The new codes perform better than the Ungerboeck codes over a wide range of SNR with SD. This is due to the fact that the Ungerboeck codes were not designed for use with sequential decoding, i.e., their distance profiles are suboptimum.
- Note that the performance of these codes with SD is only a few tenths of a dB worse than with the VA.



Comparison of New Codes with Best Known Codes

- The performance of rate 2/3 trellis codes for 8-PSK modulation using Ungerboeck codes, Porath and Aulin codes, and the new codes is compared using sequential decoding at an SNR= 7.75 dB.The new codes have the best performance over the entire range of constraint lengths.

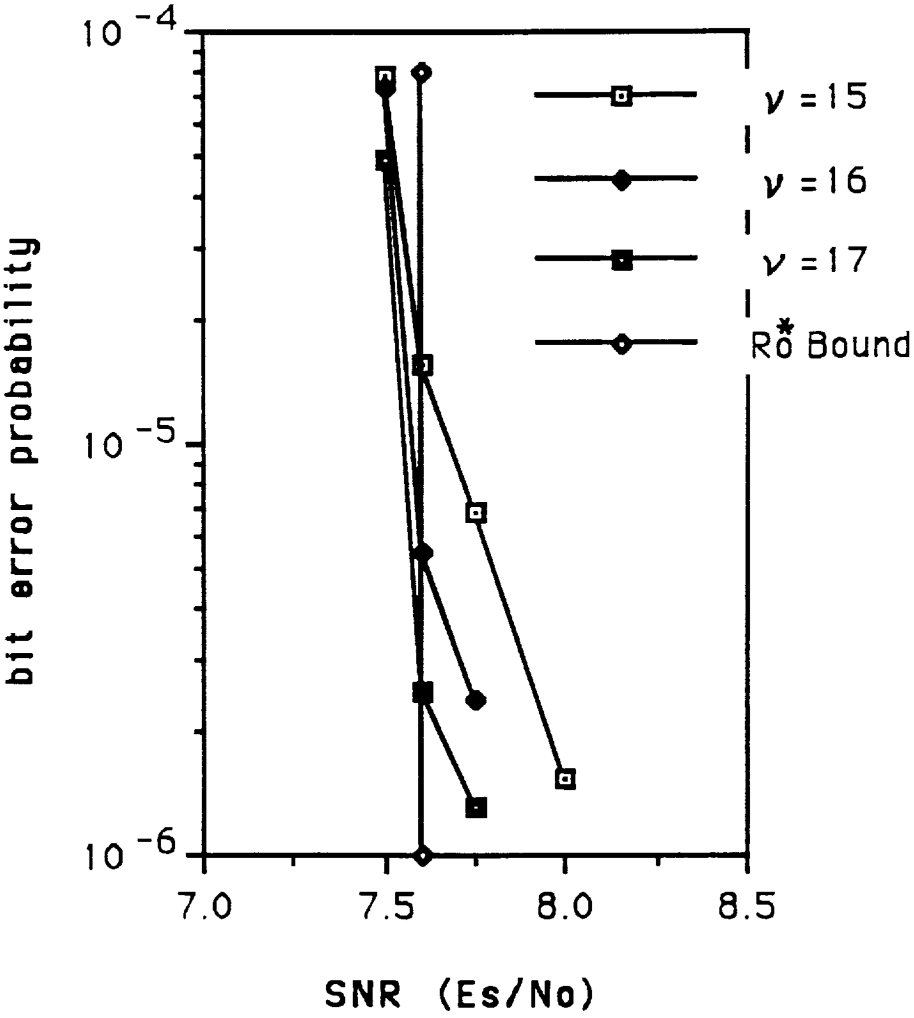
- The same approach can be used to construct rate 3/4 trellis codes for 16-QAM modulation. Similar results are obtained. However, the longest previously known 16-QAM trellis code has $\nu = 10$. We have constructed new codes for 16-QAM with ν up to 20 .



Approaching the Cut-off rate Bound for Rate 2/3 Trellis Coded 8-PSK Using Long Codes

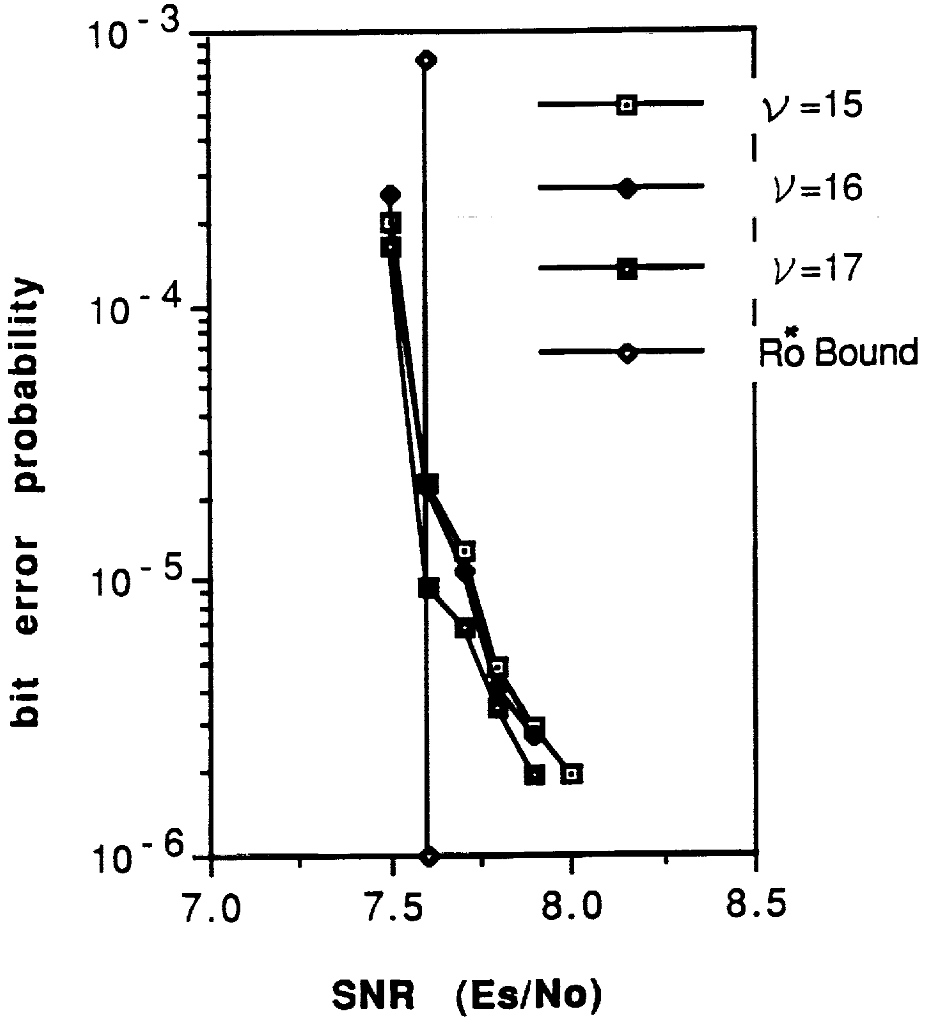
- Complete sequential decoding with an infinite input buffer is assumed.

- Our aim is to approach the R_0^* bound at a BER of $10^{-5} \sim 10^{-6}$. The performance of some new codes is shown below. Note that the cut-off rate bound is achieved at a BER of 10^{-5} with $\nu = 16$, but that larger constraint lengths will be needed to achieve the bound at a BER of 10^{-6}.



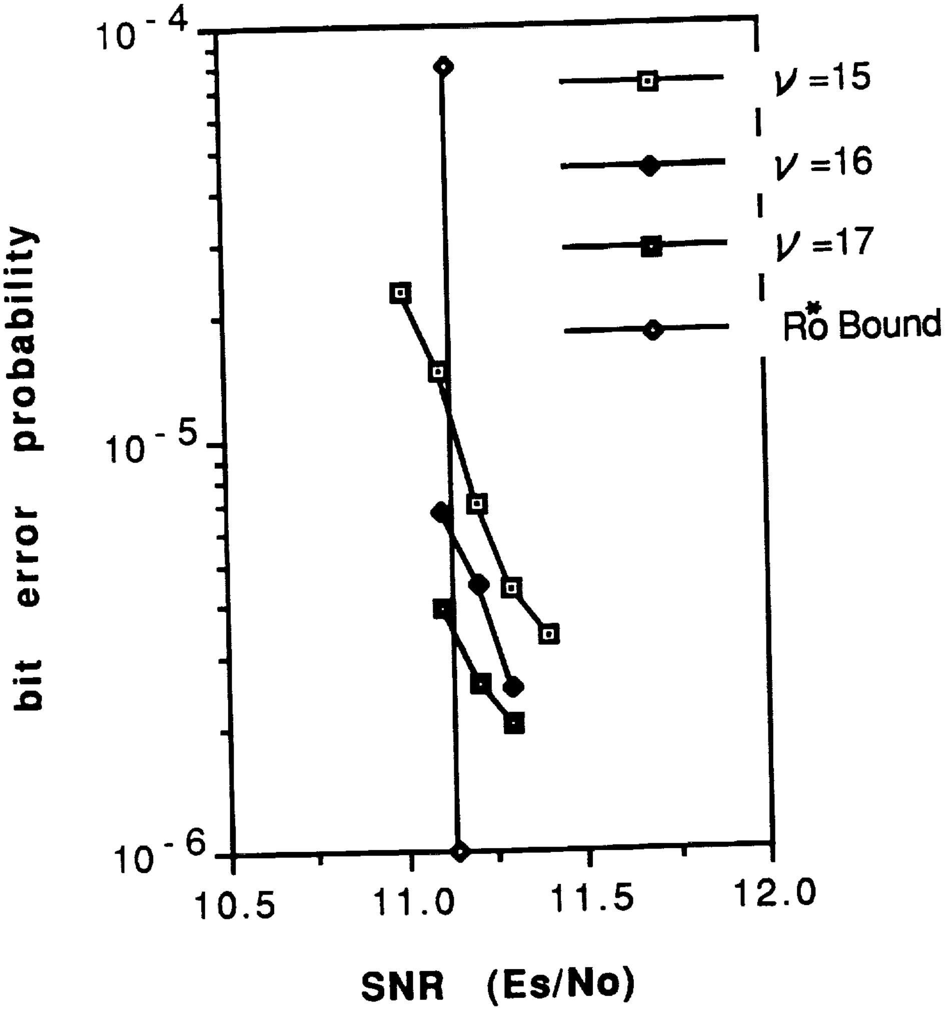
Erasurefree Sequential Decoding of Long Codes (Rate 2/3, 8-PSK)

- The Buffer Looking Algorithm (BLA) is a modification of SD which guarantees erasurefree decoding by adjusting the decoding speed before the input buffer can overflow. This results in some loss in BER performance.
- The BLA with a buffer size of 64 K symbols, a speed factor of 16, and an information block size of 512 symbols is used. Note that the cut-off rate bound is achieved at a BER of 10^{-5} with $\nu = 17$, only one larger than for the infinite buffer case.
- The BLA with a speed factor of 4 can achieve a BER of 10^{-5} at a SNR=7.8 dB, only 0.2 dB away from the cut-off rate bound.



Approaching the Cut-off Rate Bound for Rate 3/4 Trellis Coded 16-QAM Using Long Codes

- The performance of some new rate 3/4 trellis codes with 16-QAM modulation is shown below.



Computational Effort of SD: Theory

- It is well known that the average number of computations for SD has the following properties:

$$C_{av}\begin{cases} \rightarrow \infty, & R > R_0 \\ = \frac{\rho A}{\rho - 1}, & R < R_0 \end{cases}$$

where A and ρ are constants related to a specific version of sequential decoding and the code rate R, respectively.

- Note that C_{av} is independent of the code constraint length ν.

- Also note that R_0 is the maximum rate at which SD can achieve good performance.

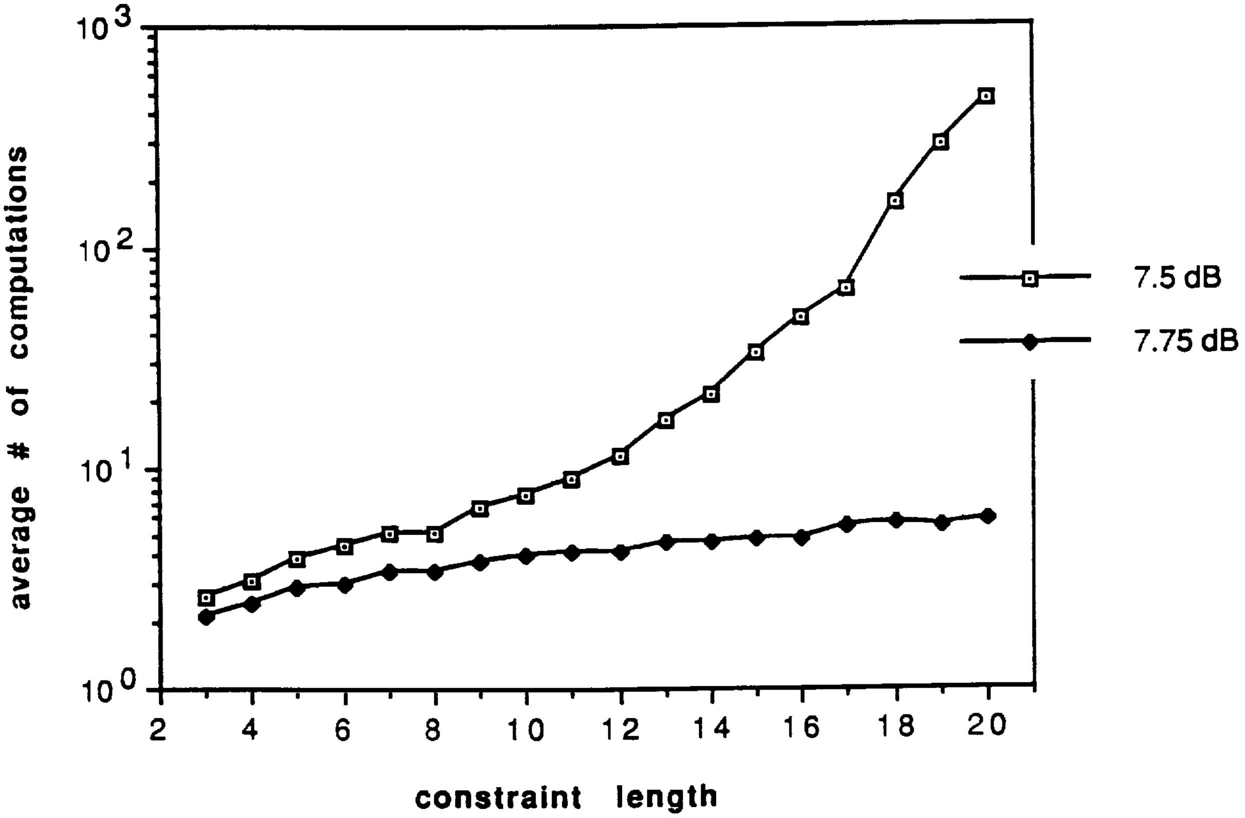
Computational Effort of SD: Practice

- The figure below shows the average number of computations for sequential decoding of trellis coded 8-PSK as a function of ν at an $SNR = 7.5dB$ (below the R_0 bound) and an $SNR = 7.75dB$ (above the R_0 bound).

- C_{av} increases moderately with increasing ν for $R < R_0$.

- C_{av} increases rapidly with increasing ν for $R > R_0$.

- This figure shows that it is not possible to beat the R_0 bound using SD.



Conclusions

• Codes which achieve the cut-off rate bound at BER's of $10^{-5} \sim 10^{-6}$ can be constructed using a random search approach. This has been demonstrated by constructing trellis codes for 8-PSK and 16-QAM modulation. Significant coding gains can be achieved when sequential decoding is used to decode those codes.

• The codes constructed in this paper outperform the best known codes using sequential decoding. Using the buffer looking algorithm, a modification of sequential decoding which eliminates erasures, our results show that the cut-off rate bound can be achieved at a BER of 10^{-5} for both 8-PSK and 16-QAM modulation. This performance is obtained using moderately large constraint lengths and reasonable decoder speed factors. Compared with Viterbi decoding of short constraint length codes, more than 1 dB of additional coding gain can be achieved at a BER of 10^{-5}.

Appendix D

On Multilevel Trellis MPSK Codes

On Multilevel Trellis MPSK Codes [*]

Jiantian Wu
Daniel J. Costello, Jr.
Lance C. Perez

Department of Electrical Engineering
University of Notre Dame
Notre Dame, Indiana 46556

Presented at the 1991
International Symposium on Information Theory
Budapest, Hungary

June 1991

[*] This work was supported in part by NASA Grant NAG5-557, NSF Grant NCR89-03429, and NSF Fellowship NSF-RCD89-54851.

Introduction

- Ungerboeck has designed two-dimensional (2D) single level trellis codes (SLTC's) for MPSK modulation with asymptotic coding gains of 3–6 dB.
- Pietrobon, Deng, et. al. have designed multi-dimensional (MD) SLTC's for MPSK modulation with comparable coding gains and higher spectral efficiencies.
- These SLTC's use Viterbi decoding and are found by exhaustive search. The decoding and code search complexity both grow exponentially with constraint length.
- Thus, it is very difficult to find and optimumly decode SLTC's with large Euclidean distance.
- Multilevel trellis codes (MLTC's) with large Euclidean distance are easily designed and can be decoded using suboptimum reduced complexity multistage decoding (MSD). (*Calderbank, Sayegh*)
- *Problem:* Can one design MLTC's with a clear performance-complexity advantage over SLTC's? There are two difficulties:

  1. High path multiplicities (error coefficients) of MLTC's.
  2. Performance loss due to MSD.

Coded Modulation Performance

- One measure of the performance of coded modulation schemes is the asymptotic coding gain

$$\gamma = 10 \log_{10} \left(\frac{d_f^2}{d_u^2} \right)$$

  where d_f^2 is the minimum squared Euclidean distance (MSED) of the coded system and d_u^2 is the MSED of an uncoded system with the same spectral efficiency.

- For 2D SLTC's, simulation results have shown that the real coding gain is within $0.5 - 1.5$ dB of the asymptotic coding gain at a bit error rate (BER) of 10^{-5}. Thus, γ is a reasonable measure of the code performance.

- For MD SLTC's and for MLTC's, there can be a much larger difference between the real and asymptotic coding gains at 10^{-5} due to the dense distance spectra and high path multiplicities.

- Hence, it is necessary to develop a better analytical measure of performance for these codes.

- We have treated the real coding gain of MD SLTC's in a previous paper. In this paper, we examine the real performance (real coding gain) of 2D MLTC's with MPSK modulation using binary partitions.

MPSK MLTC's

- A 2D MPSK signal set can be refined into $\log_2(\mathrm{M})$ levels using a binary partition. This induces a mapping of $\log_2(\mathrm{M})$-tuples to individual points in the MPSK signal set assuming natural labeling.

- With MLTC's a different code is used at each of the $\log_2(\mathrm{M})$ levels in the binary partition. Some of the levels may be left uncoded.

- The codes at each level are called *Component Codes* and are denoted C_i for $i = 1, 2, \ldots, \log_2(\mathrm{M})$.

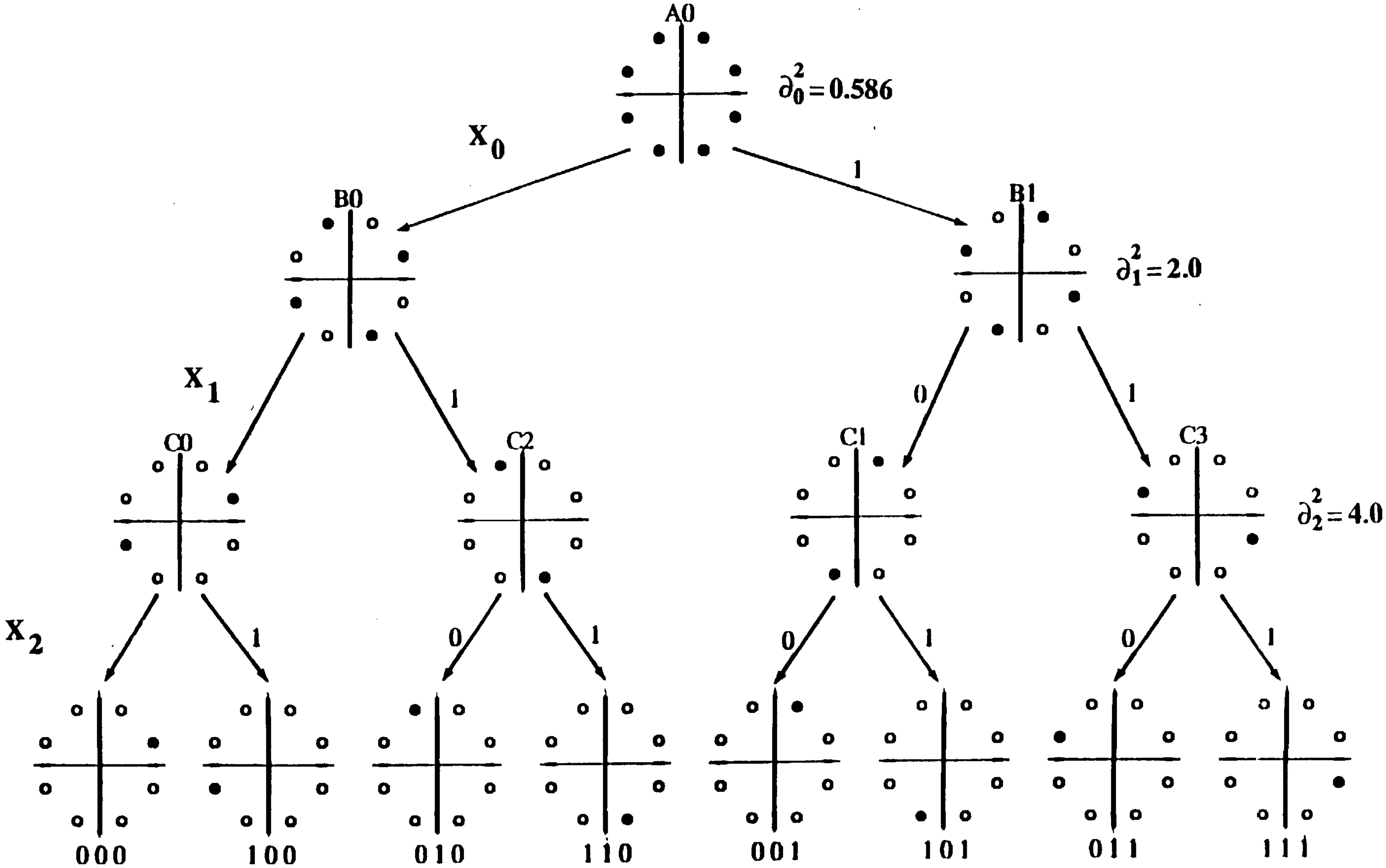
- For a $\log_2(\mathrm{M})$ level MPSK trellis code with component codes C_i, the minimum squared Euclidean distance is given by

$$d_f^2 = min\{d_i \delta_i^2, i = 1, 2, \ldots, \log_2(\mathrm{M})\}$$

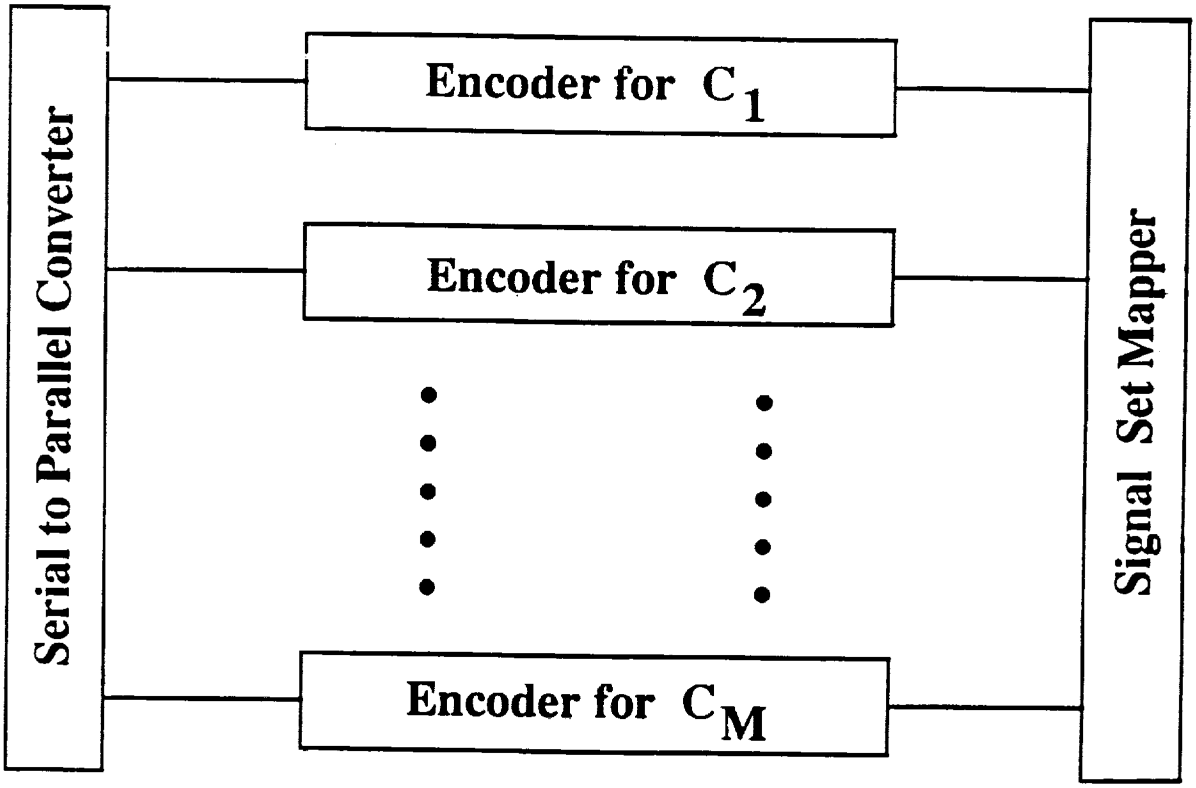
  where d_i is the minimum Hamming distance of the i^{th} component code and δ_i^2 is the minimum squared intrasubset distance at the i^{th} level in the binary partition of the signal set. (*Ginzburg, Tanner*)

- This usually leads to choosing C_1 to have a large minimum Hamming distance.

8PSK Binary Partition



General MLTC Encoder Block Diagram



Example 1

- Consider a three level 8PSK trellis code with

  1. C_1 an 8 state rate 1/4 convolutional code with $d_1 = 13$
  2. C_2 an 8 state rate 3/4 convolutional code with $d_2 = 4$
  3. C_3 a $(32, 31)$ single parity check block code with $d_3 = 2$

  and rate $R = 1/4 + 3/4 + 31/32 = 1.97$ bits/symbol.

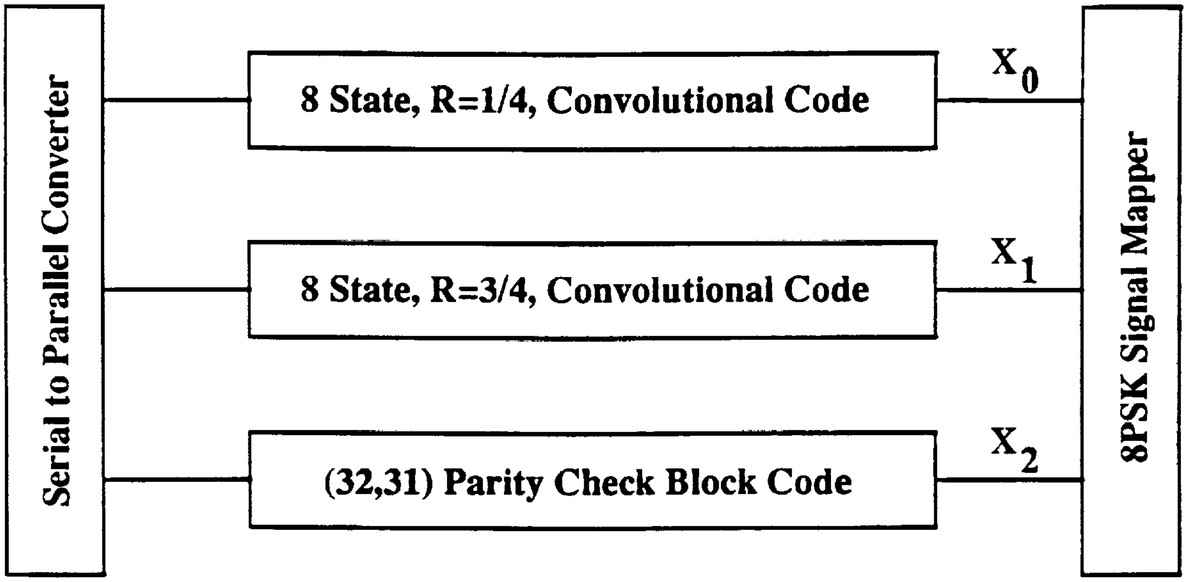
- This code has free distance

$$\begin{aligned} d^2_{free} &= min\{13\text{x}0.586, 4\text{x}2, 2\text{x}4\} \\ &= 7.618 \end{aligned}$$

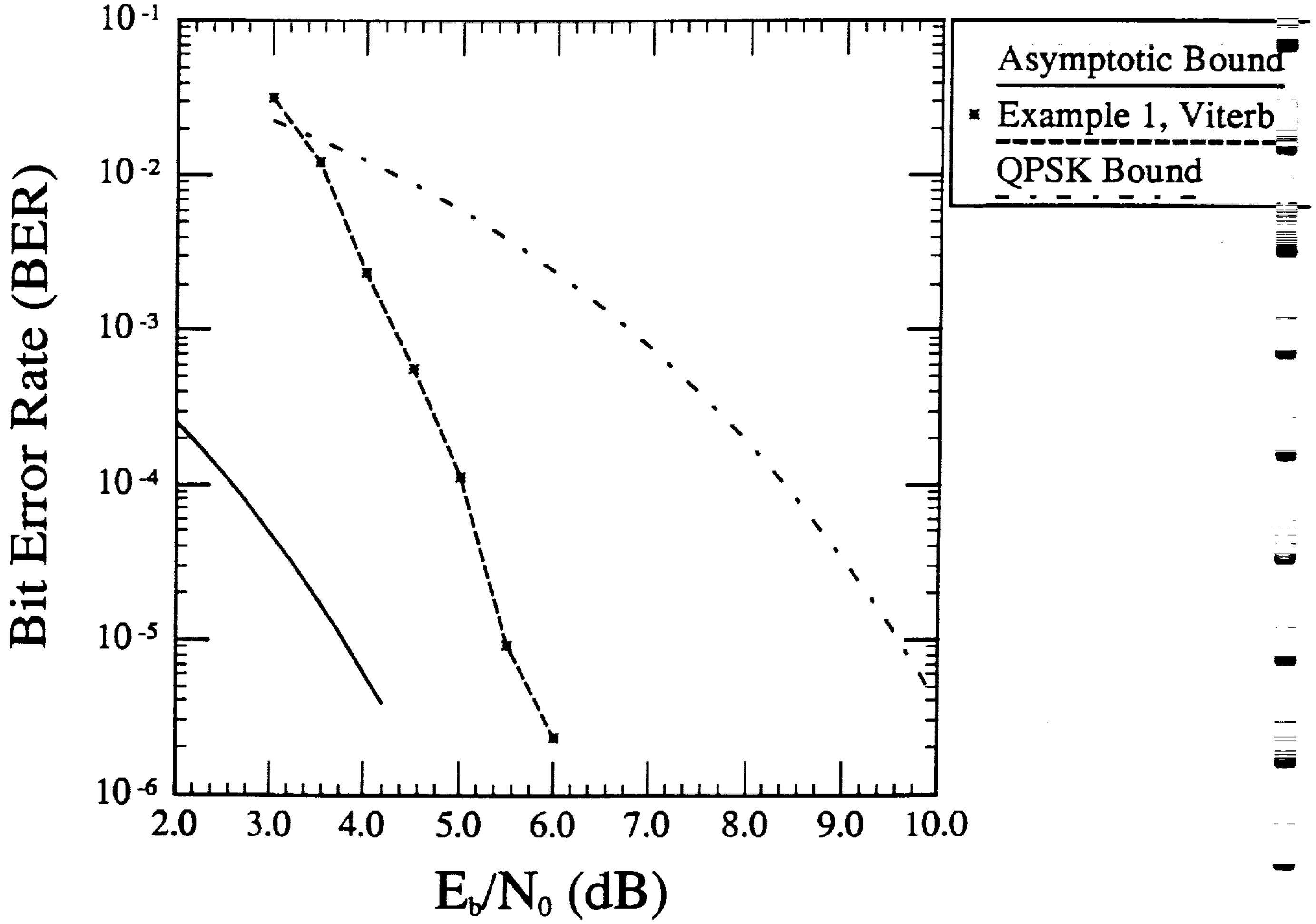
and an asymptotic coding gain of

$$\gamma = 5.81\text{dB}$$

relative to uncoded 4PSK ($d^2_u = 2$).


Serial to Parallel Converter
8 State, R=1/4, Convolutional Code
8 State, R=3/4, Convolutional Code
(32,31) Parity Check Block Code
X0
X1
X2
8PSK Signal Mapper

Simulation Results



An Argument for the Gap at 10^{-5}

- Forney has proposed a rule of thumb for trellis coded modulation schemes that states that each increase in path multiplicity by a factor of 2 results in a 0.2dB loss in real coding gain at 10^{-5}.
- Thus, the *effective coding gain* of a trellis code at 10^{-5} is given by

$$\gamma_{eff} = 10 \log_{10}\left(\frac{d_f^2}{d_u^2}\right) - 0.2 \log_2\left(\frac{N_{free}}{2}\right)$$

  where N_{free} is the number of paths at the code's free distance. (N_{free} of any uncoded 2D MPSK system is 2.)

- Since there are two nearest neighbors in each subset at the first level of a binary MPSK partition, the MLTC has

$$N_{eff} = 2^{d_1} N(d_1)$$

  paths at distance $d_1\delta_1^2$ due to the first component code, where $N(d_1)$ is the multiplicity of the minimum distance path of C_1. Subsequent component codes may also affect the path multiplicity.

- N_{eff} is called the *effective path multiplicity.*
- For 8PSK, assuming that $d_1\delta_1^2 \leq d_2\delta_2^2$ and $d_1\delta_1^2 \leq d_3\delta_3^2$, then

$$\gamma_{eff} = 10 \log_{10}\left(\frac{d_f^2}{d_u^2}\right) - 0.2 \log_2\left(\frac{2^{d_1}N(d_1)}{2}\right)$$

  which reaches a maximum of 3.89 dB for $d_1 = 22$ and $N(d_1) = 1$.

Example 1 Revisited

- For the MLTC of Example 1, $d_1 = 13$ and $N(d_1) = 1$. Thus, the effective path multiplicity due to the first component code is $N_{eff} = 2^{13}$ and the effective coding gain at 10^{-5} is only

$$\gamma_{eff} = 3.41\text{dB},$$

or 2.4dB less than γ.

Example 2

- If we change C_1 to the 16 state rate 1/4 convolutional code with $d_1 = 16$ and $N(d_1) = 1$, then $d_{free}^2 = 8.0$ and

$$\gamma = 6.02\text{dB},$$

but $N_{eff} = 2^{16}$ and the effective coding gain is only

$$\gamma_{eff} = 3.71\text{dB}.$$

- Conclusions (based on Forney's rule of thumb):

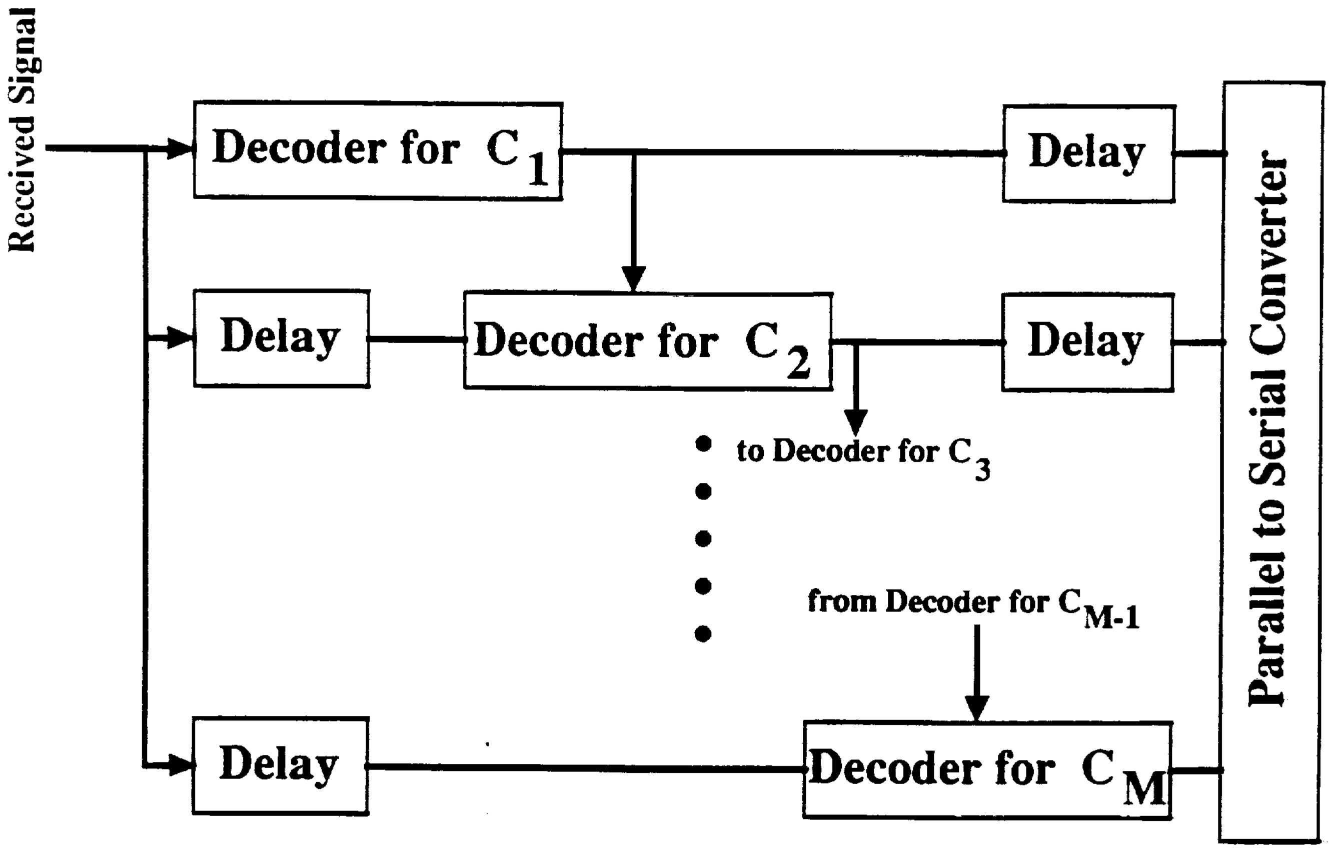
1. For MLTC's increasing the MSED involves using low rate codes with large minimum distance at the first level. This results in an exponentially increasing effective multiplicity.
2. It is unlikely that MLTC's can achieve more than 4dB of real coding gain at a BER of 10^{-5}.

Performance Loss Due to Multistage Decoding

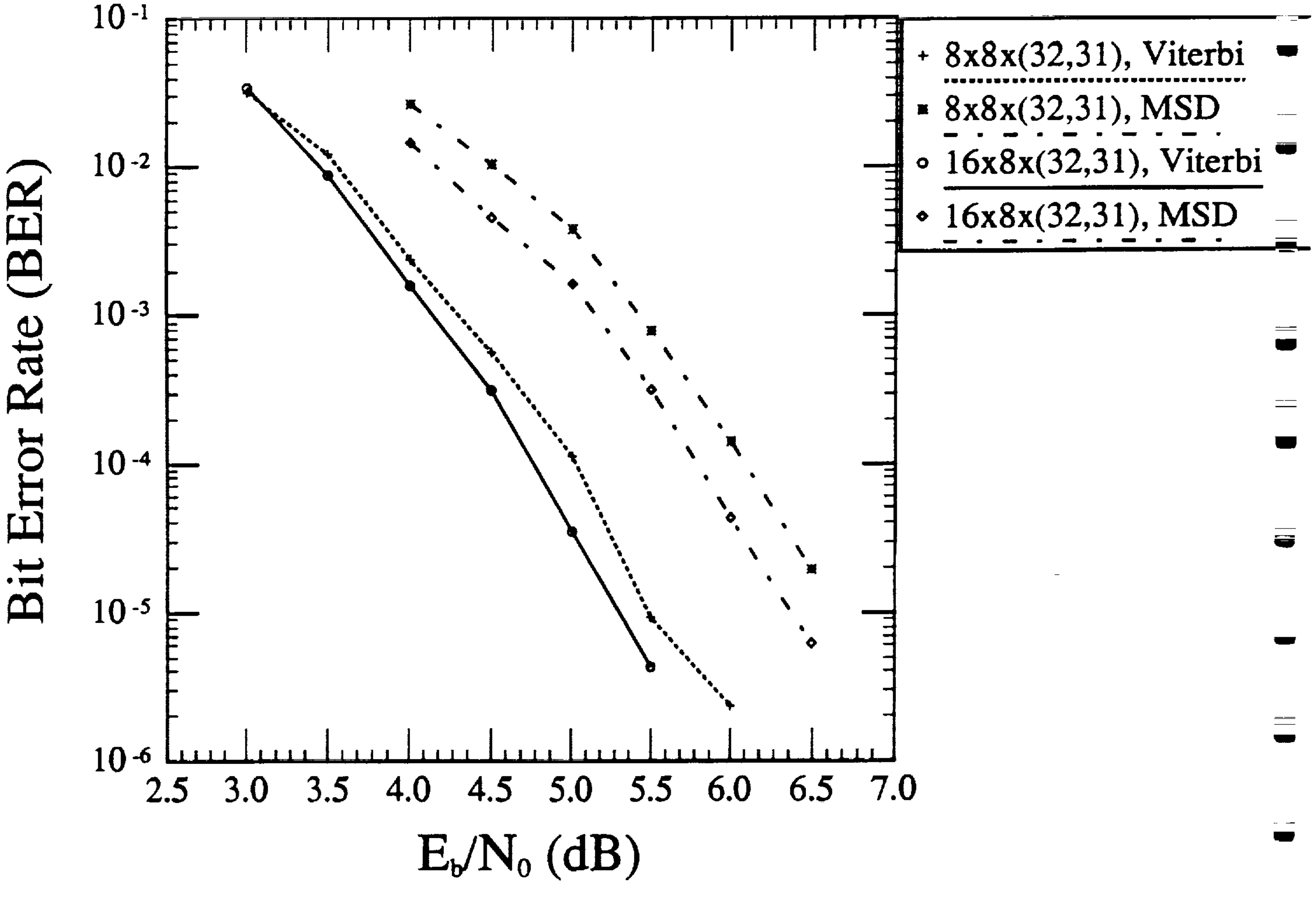
- Another issue in the performance of MLTC's is the suboptimality of multistage decoding (MSD). That is, how much coding gain is lost due to MSD?

- The loss due to MSD can be measured by comparing simulation results of a specific MLTC using maximum likelihood Viterbi decoding and MSD.

- For the multilevel codes in examples 1 and 2, simulation results show a loss of 1dB at a BER of 10^{-5} due to MSD. This is comparable to the loss of other suboptimal decoding techniques relative to Viterbi decoding.



Simulation Results



Comparison to SLTC's

- The $R = 2/3$ (2 bits/symbol), 16 state, 8PSK Ungerboeck SLTC is of comparable complexity to the MLTC of Example 1, but has an asymptotic coding gain of

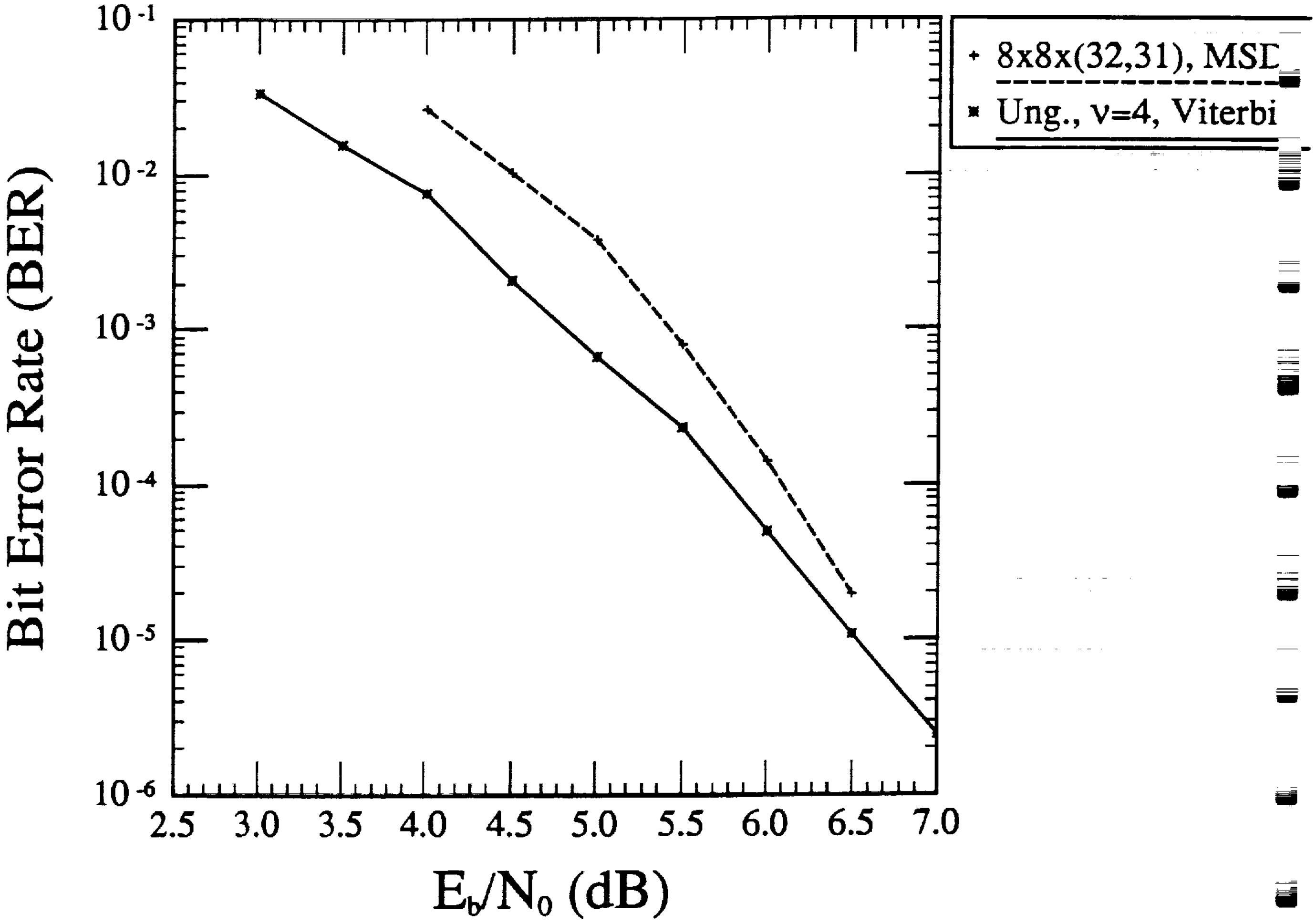
$$\gamma = 4.13\text{dB}$$

and a real coding gain (from simulations) of 3.2 dB at a BER of 10^{-5}.

- Conclusions:

1. Most MLTC's with large MSED's will not have a performance-complexity advantage over SLTC's in terms of real coding gain at a BER of 10^{-5}.
2. However, the asymptotic coding gain of MLTC's is superior to that of SLTC's.
3. It is possible to construct high rate, two level MLTC's with a clear performance-complexity advantage over comparable SLTC's.

Simulation Results



High Rate MPSK MLTC's

- By using high rate codes at the first level, it is possible to construct some high rate MLTC's with a performance-complexity advantage over high rate MD SLTC's. (*This has also been suggested by Pottie and Taylor.*)
- The use of high rate (low minimum distance) codes for C_1 reduces the high multiplicities of MLTC's.
- With

  1. C_1 a rate 1/2 convolutional code,
  2. C_2 an $(L, L-1)$ single parity check block code with $d_2 = 2$,
  3. C_3 uncoded,

  we can construct a class of rate

$$R = 1/2 + (L-1)/L + 1 = 2.5 - \frac{1}{L}\text{bits/symbol}$$

MLTC's with a performance-complexity advantage over the $R = 2.5$ bits/symbol, 2x8PSK codes of Pietrobon, Deng, et. al..

Example 3

- **Consider a three level 8PSK trellis code with**

  1. C_1 **a 4 state rate 1/2 convolutional code with** $d_1 = 5$,
  2. C_2 **a** $(64, 63)$ **single parity check block code with** $d_2 = 2$,
  3. C_3 **uncoded,**

  and rate $R = 1/2 + 63/64 + 1 = 2.48$ **bits/symbol.**

- **This code has free distance**

$$\begin{aligned} d^2_{free} &= min\{5x0.586, 2x2, 1x4\} \\ &= 2.93, \end{aligned}$$

  with $N_{eff} = 32$, **an asymptotic coding gain of**

$$\gamma = 3.98\text{dB},$$

  and an effective coding gain of

$$\gamma_{eff} = 3.18\text{dB}.$$

- **The** $R = 2.5$ **bits/symbol, 4 state, 2x8PSK SLTC of Pietrobon, Deng, et. al. has**

$$d^2_{free} = 2.0,$$

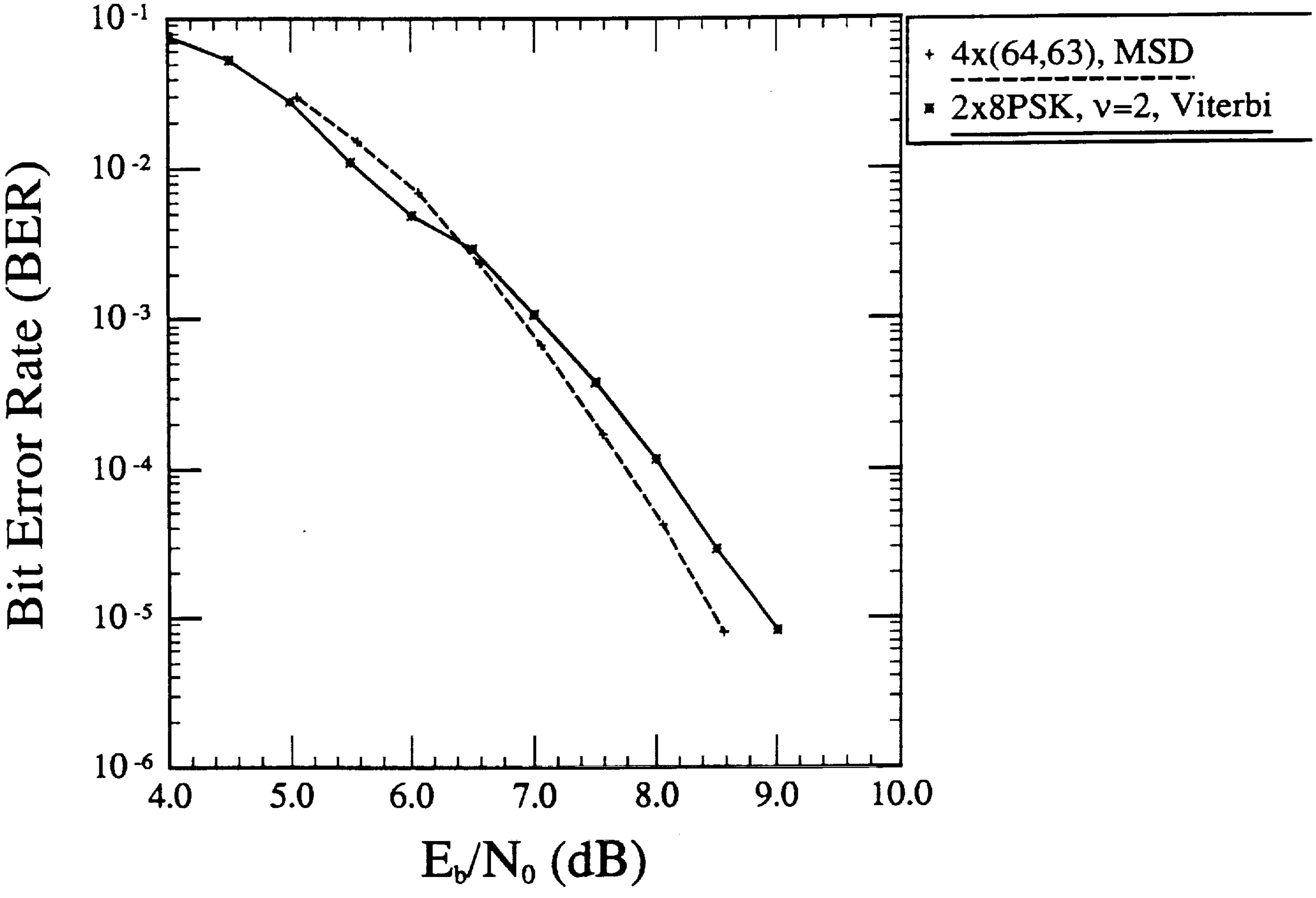
  with $N_{free} = 4$, **an asymptotic coding gain of**

$$\gamma = 2.32\text{dB},$$

  and a real coding gain (from simulations) of 2.3 **dB at a BER of** 10^{-5}.

- **Note,** $d_u^2 = 1.172$ **and** $N_u = 4$, **so** $\gamma_{eff} = \gamma$.

Simulation Results



Concatenated Multilevel MPSK Codes

- Part of the performance loss of MLTC's with MSD is the result of error propagation from previously decoded higher levels to lower levels. It makes sense, then, to try to prevent errors at the higher levels.

- One way of combating error propagation is to use concatenated coding on one or more levels. *(This has also been suggested by Rajpal, Rhee, and Lin and by Herzberg, Be'ery, and Snyders.)*

- Let $P_{i,in}$ and $P_{i,out}$ denote the probability of incorrectly decoding the i^{th} stage inner and outer code, respectively, and $P_{i,in|1,out,\ldots,i-1,out}$ the probability of incorrectly decoding the i^{th} stage inner code given that the first $(i-1)$ outer codes were correctly decoded.

- Let $P_{2,in|1,out,correct}$ and $P_{2,in|1,out,incorrect}$ be the probability of incorrectly docoding the second level inner code given that the first outer code was correctly and incorrectly decoded, respectively. Then,

$$P_{2,in} = (1 - P_{1,out})P_{2,in|1,out,correct} + P_{1,out}P_{2,in|1,out,incorrect},$$

  where the last term represents the error propagation from the first level to the second level.

- Since the first outer code is very powerful,

$$P_{1,out} \ll 1$$

  and

$$P_{1,out} \ll P_{2,in|1,out,correct}$$

  and the expression for $P_{2,in}$ is dominated by the first term.

- Thus,

$$P_{2,in} \approx P_{2,in|r_1,out,correct}$$

  and the error propagation has essentially disappeared.

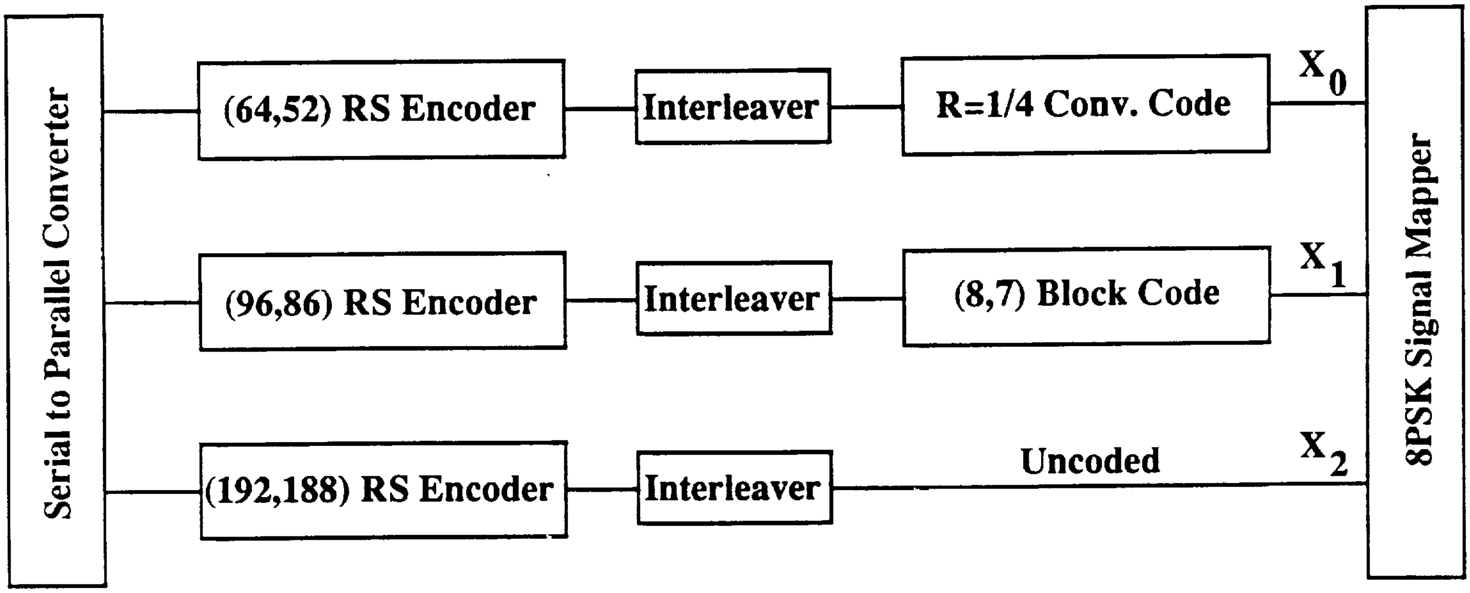
Example 4

- For example, consider a three level 8PSK trellis code with

  1. $C_{1,in}$ a 4 state rate 1/4 convolutional code,
  2. $C_{2,in}$ an $(8,7,2)$ single parity check block code,
  3. $C_{3,in}$ uncoded,

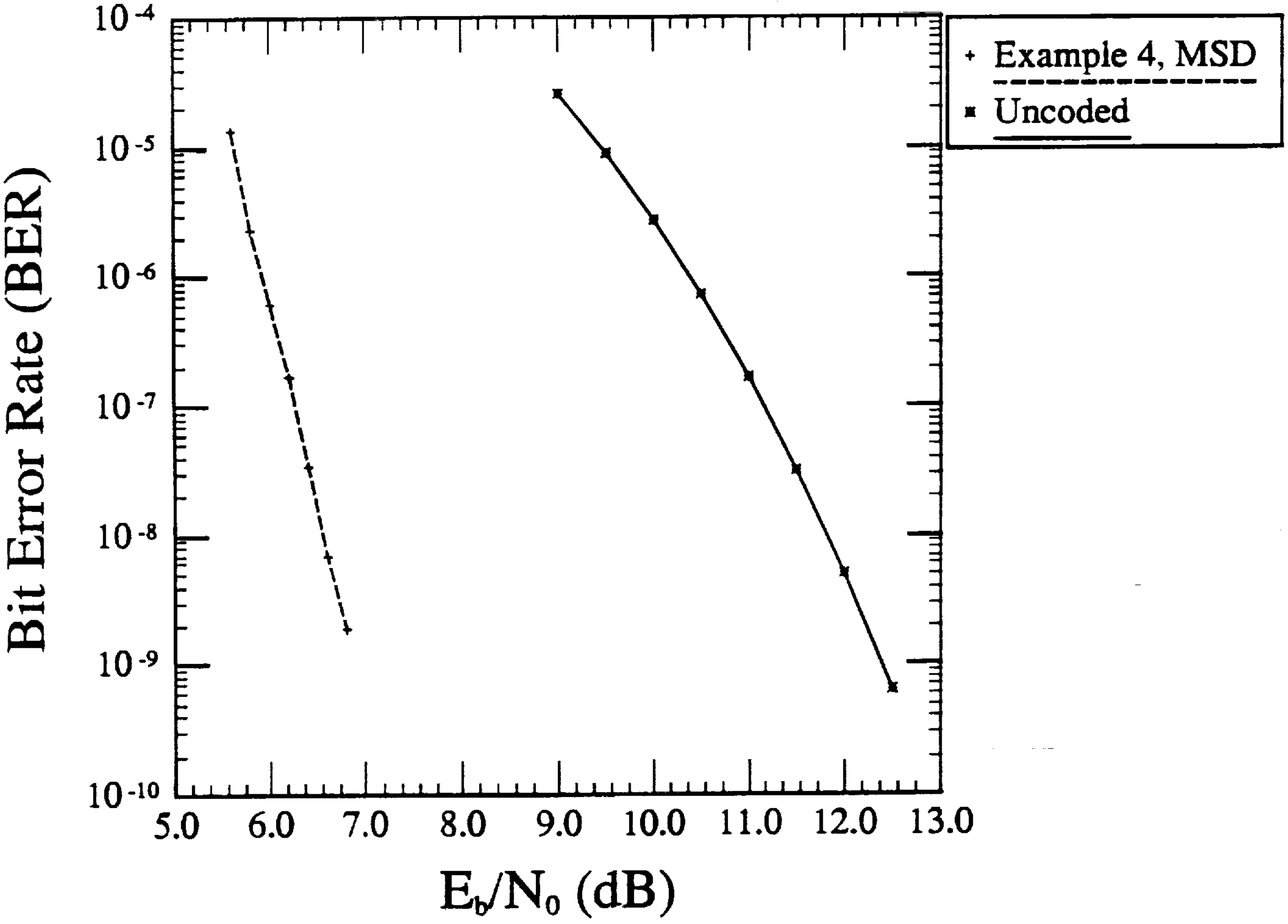
  and

  1. $C_{1,out}$ the extended $(64,52)$ RS code,
  2. $C_{2,out}$ the shortened $(96,86)$ RS code,
  3. $C_{3,out}$ the shortened $(192,188)$ RS code.



- This code has a spectral efficiency of 1.966 bits/symbol and simulation results show a real coding gain of 4.0dB at a BER of 10^{-5}.

Simulation Results



Conclusions

- The dense distance spectra and high path multiplicities of multilevel trellis codes result in a significant difference between the asymptotic and real coding gains.
- The effective path multiplicity can be used to estimate the real coding gain of multilevel trellis codes at a BER of 10^{-5}.
- The performance loss due to MSD is comparable to that of other suboptimal decoding techniques.
- High rate, 2 level MLTC's with MSD appear to have a distinct performance-complexity advantage compared to SLTC's with Viterbi decoding.
- Error propagation in MSD can be effectively reduced by using concatenated codes at one or more levels.
- It may be possible to reduce the effective multiplicity of MLTC's with MSD by introducing dependencies between levels. However, this makes finding the free distance of MLTC's more difficult. (*Ginzburg and Tanner's bound may no longer hold.*)

Appendix E

M-Algorithm Decoding of the Proposed Nonlinear 64-State V.FAST Code

M-algorithm Decoding of the Proposed Nonlinear 64-State V.FAST Code [*]

Lance C. Perez
Daniel J. Costello, Jr.

Department of Electrical and Computer Engineering
University of Notre Dame
Notre Dame, Indiana 46556

May 7, 1991

Abstract

M-algorithm decoding of a rotationally invariant nonlinear 64 state trellis code proposed for the CCITT V.FAST "ultimate modem" standard is considered. In this study, the M-algorithm is implemented in a continuous mode of operation, that is, the data is not framed. Simulation results show that M = 16 gives performance as good as or better than the best known 16 state code with Viterbi decoding at a bit error rate of 10^{-5}. When compared to Viterbi decoding of the 64 state nonlinear code, M = 16 loses less than 0.5dB and M = 8 less than 1.2dB at 10^{-5}. Asymptotically, the M-algorithm performance approaches that of the 64 state Viterbi decoder even for small M.

[*] This work was supported in part by NASA Grant NAG5-557, NSF Grant NCR89-03429, and NSF Fellowship NSF-RCD89-54851.

1 Introduction

The complexity of a rotationally invariant nonlinear 64 state trellis code[1] proposed for the CCITT V.FAST "ultimate modem" standard has led to the consideration of suboptimal decoding techniques for initial implementation. An appropriate suboptimum decoder should have the following properties:

- Significant reduction in complexity compared to a full 64 state Viterbi decoder.
- Fixed decoding delay.
- Fixed number of computations per decoded branch.
- Not require framing of the information bits.
- Minimal reduction in the effective real coding gain and asymptotic performance approaching that of the full 64 state Viterbi decoder.

A decoding technique that appears to satisfy these criteria is the M-algorithm.

2 The M-algorithm

The M-algorithm is a reduced state trellis search decoding algorithm parameterized by M, the number of states or paths stored by the decoder, and L, the decoder path memory (truncation length)[2-4]. The algorithm consists of the following three steps:

1. Path extension
2. Path deletion
3. Sorting.

It is best explained in terms of an example.

Consider the nonlinear 64 state trellis code using 16QAM modulation from [1] and M-algorithm decoding with M = 8 and L = 20. With this signal set, the code has rate R = 3/4 with $\tilde{m} = 2$ coded information bits and $m - \tilde{m} = 1$ uncoded bit resulting in 2 parallel transitions. At time nT, the decoder has stored M = 8 paths, their metrics, and the path history for the last 20 branches. It is important to recognize that each of the M = 8 paths terminates in a unique state (though it may be any of the 64 states). At time $(n+1)T$, the decoder performs the following operations:

- Extends each of the M = 8 stored paths into $2^{\tilde{m}} = 4$ paths and computes each of the 4M = 32 new path metrics. Subset (parallel transition) decoding is assumed to be done by the demodulator.
- If any of the 4M = 32 paths now end in the same state (at time nT the M = 8 paths ended in unique states), then only the path with the best metric for each state is retained and the rest are deleted. This is essentially an Add-Compare-Select(ACS) operation.
- The remaining paths are then sorted by path metric and the best M = 8 are chosen as survivors. The oldest branch associated with the current best path is released as decoded information bits and the newly decoded branches are stored in the path memory.

For M much smaller than the full number of states, path deletion (Step 2) is not often required and is ignored in complexity considerations. Nevertheless, the path deletion step is important to the M-algorithm and performance degrades if it is ignored.

Ignoring the path deletion step, the M-algorithm requires MB addition operations, where B $\triangleq 2^{\tilde{m}}$, and the sorting operation. It can be shown that finding the best M paths out BM paths requires

$$C_M \leq BM + M + f(BM)$$

computations, where $f(BM)$ is a function that grows more slowly than its argument, that is

$$\lim_{BM \to \infty} \frac{f(BM)}{BM} = 0.$$

The computational complexity of the M-algorithm is then $M(2B+1) + f(BM)$. A Viterbi decoder with V states requires BV addition operations and (B-1)V compare operations for a total of $(2B-1)V$ computations. Thus, if M is much less than V, the M-algorithm results in a significant reduction in complexity. It is also clear that the decoding delay and the number of computations per decoded branch are fixed. With M=V, the M-algorithm is identical to the Viterbi algorithm.

The principal difficulty with the M-algorithm is the possibility that at some time the correct path is not among the best M paths and is discarded. This is referred to as path loss and leads to long bursts of errors until the decoder reacquires the correct path. To mitigate this effect, the information stream is usually framed and the decoder periodically forced to the all zero state[5]. Since framing is undesirable in the V.FAST application, this approach is not taken here and the M-algorthm is operated in a continuous mode with no framing. It should be noted that there exist a number of path recovery schemes for the M-algorithm that may be used to resynchronize the decoder in place of framing[6,7]. The application of these recovery schemes to trellis codes has not been investigated.

3 Simulation Results

The performance of the M-algorithm in decoding the nonlinear 64 state code was investigated using Monte Carlo simulation. In all the simulations, unquantized squared Euclidean distance is the metric and no framing is used. The channel is simulated with an Additive White Gaussian Noise (AWGN) model. To account for the nonlinearity of the proposed code and the nonuniform error probability of trellis codes in general, the performance was averaged over four nonzero information sequences.

In Figure 1, the performance of the M-algorithm with (M,L) = (16, 120) and (8, 120) is compared to the performance of the proposed nonlinear 64 state code with Viterbi decoding and the best known linear 16 state code [8] with Viterbi decoding. A truncation length of 120 branches was used for the Viterbi decoders in this case to assure maximum likelihood decoding. At a bit error rate (BER) of 10^{-5}, the performance with (M,L) = (16, 120) is only 0.5dB worse than the 64 state Viterbi decoder and is essentially the same as that of the 16 state code. The performance of the M-algorithm with (M,L) = (8, 120) is within 1.2dB of the 64 state code and 0.6dB of the 16 state code.

Figure 2 shows the performance of the M-algorithm as a function of the path memory, L, for M = 16 and M = 8. This figure clearly demonstrates the robustness of the M-algorithm to variations in L. The performance of the nonlinear code with 64 state Viterbi decoding is shown in Figure 3 for a number of truncation lengths. There is a slight degradation in performance as the truncation length decreases.

As mentioned previously, no framing or path recovery scheme was considered in the simulation of the M-algorithm. Consequently, it is expected that path loss would occasionally occur and cause long bursts of errors. One indication of this is the variation in performance of the different information sequences relative to the average performance. This is shown in Figure 4 for M = 16 and L = 30 (the variation is also insensitive to L). This figure shows the increased variation in performance with increasing SNR that is symptomatic of path loss. However, it must be remembered that this code does not have the uniform error probability property and some variation is expected even with full 64 state Viterbi decoding. (This variation is shown in Figure 5.) Comparison of these two figures reveals only a moderate increase in variance for the M-algorithm. Though these results are far from conclusive, they indicate that path loss may not be a significant problem and that the M-algorithm can operate satisfactorily in a continuous mode with trellis codes.

4 Conclusion

The M-algorithm appears to be a promising candidate as a suboptimal decoder for the proposed nonlinear 64 state CCITT V.FAST code. The M-algorithm gives good performance with moderate complexity and has the desirable qualities of fixed delay and computation. One possible obstacle to using the M-algorithm is path loss and the resulting burst of errors. It may be possible to implement a path recovery scheme to alleviate this problem with only a moderate increase in complexity.

References

[1] S. S. Pietrobon, G. Ungerboeck, and D. J. Costello, Jr., "Rotationally Invariant Trellis Codes," *submitted to the IEEE Trans. Inform. Theory.*

[2] F. Jelinek and J. B. Anderson, "Instrumentable Tree Encoding of Information Sources," *IEEE Trans. Inform. Theory.* **IT-17**, pp. 118–119, January 1971.

[3] J. B. Anderson and C. -F. Lin, "M-Algorithm Decoding of Channel Codes," *Proc. Thirteenth Biennial Symp. on Comm.*, pp. A3.1–A3.4, Queens University, Kingston, Canada, June 2-4, 1986.

[4] C. -F. Lin and J. B. Anderson, "M-Algorithm Decoding of Channel Convolutional Codes," *Proc. of Princeton Conference on Information Science and Systems*, pp. 362–366, Princeton, New Jersey, March 1986.

[5] G. J. Pottie and D. P. Taylor, "A Comparison of Reduced Complexity Decoding Algorithms for Trellis Codes," *submitted to the IEEE Journal on Selected Areas in Communicatons.*

[6] U. Roethlisberger, *A Path Recovery Scheme for the Truncated Viterbi Algorithm*, Masters Thesis, Rensselaer Polytechnic Institute, Troy, NY, 1988.

[7] C. -F. Lin, *A Truncated Viterbi Algorithm Approach to Trellis Codes*, Ph.D. Dissertation, Rensselaer Polytechnic Institute, Troy, NY, 1986.

[8] S. S. Pietrobon, *Trellis Coding with Multidimensional Signal Sets and Rotationally Invariant Trelli Codes"*, Ph.D. Dissertation, Univ. of Notre Dame, Notre Dame, IN., 1990.

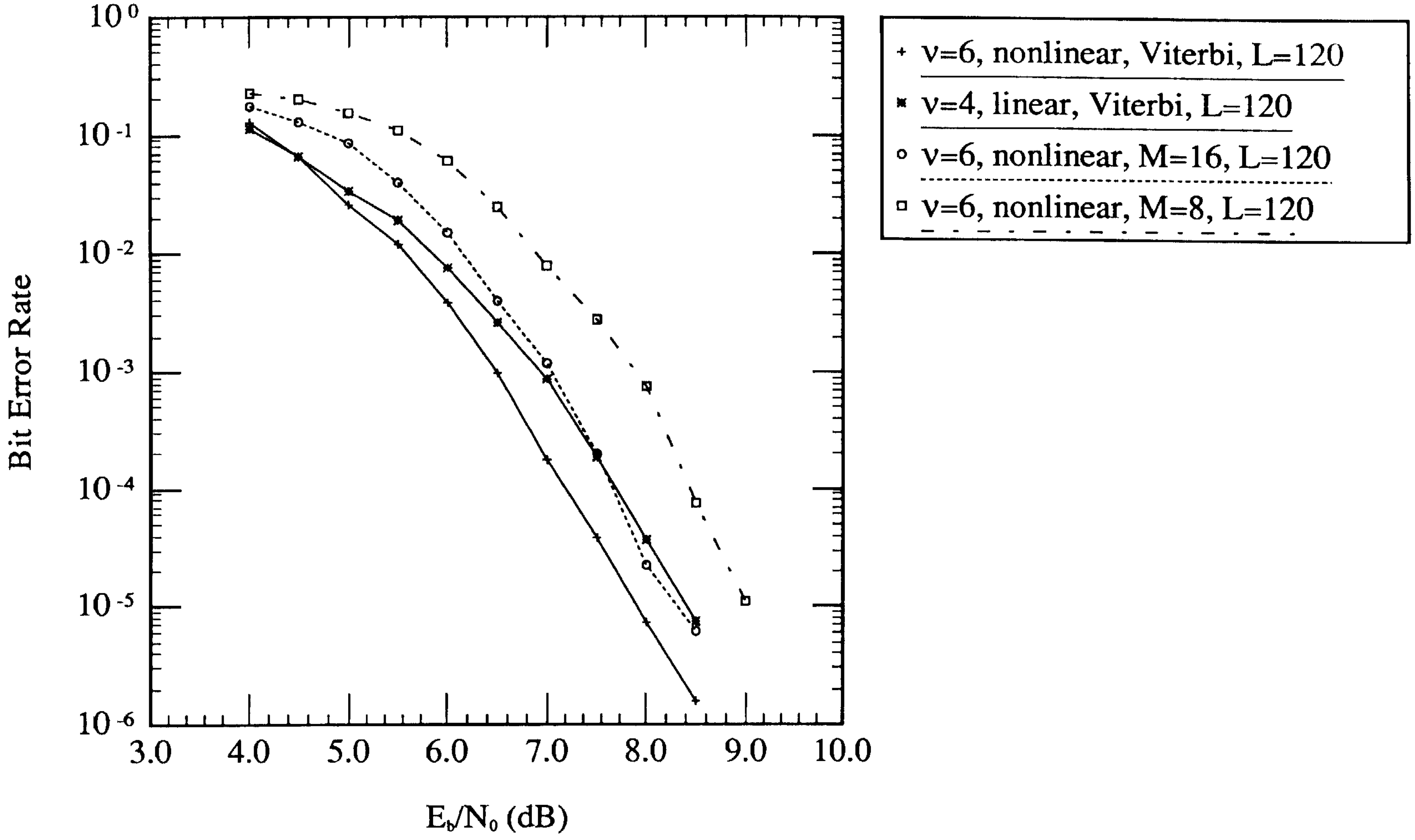


Figure 1: Simulation Results for the Nonlinear, 64 State, 16QAM Code.

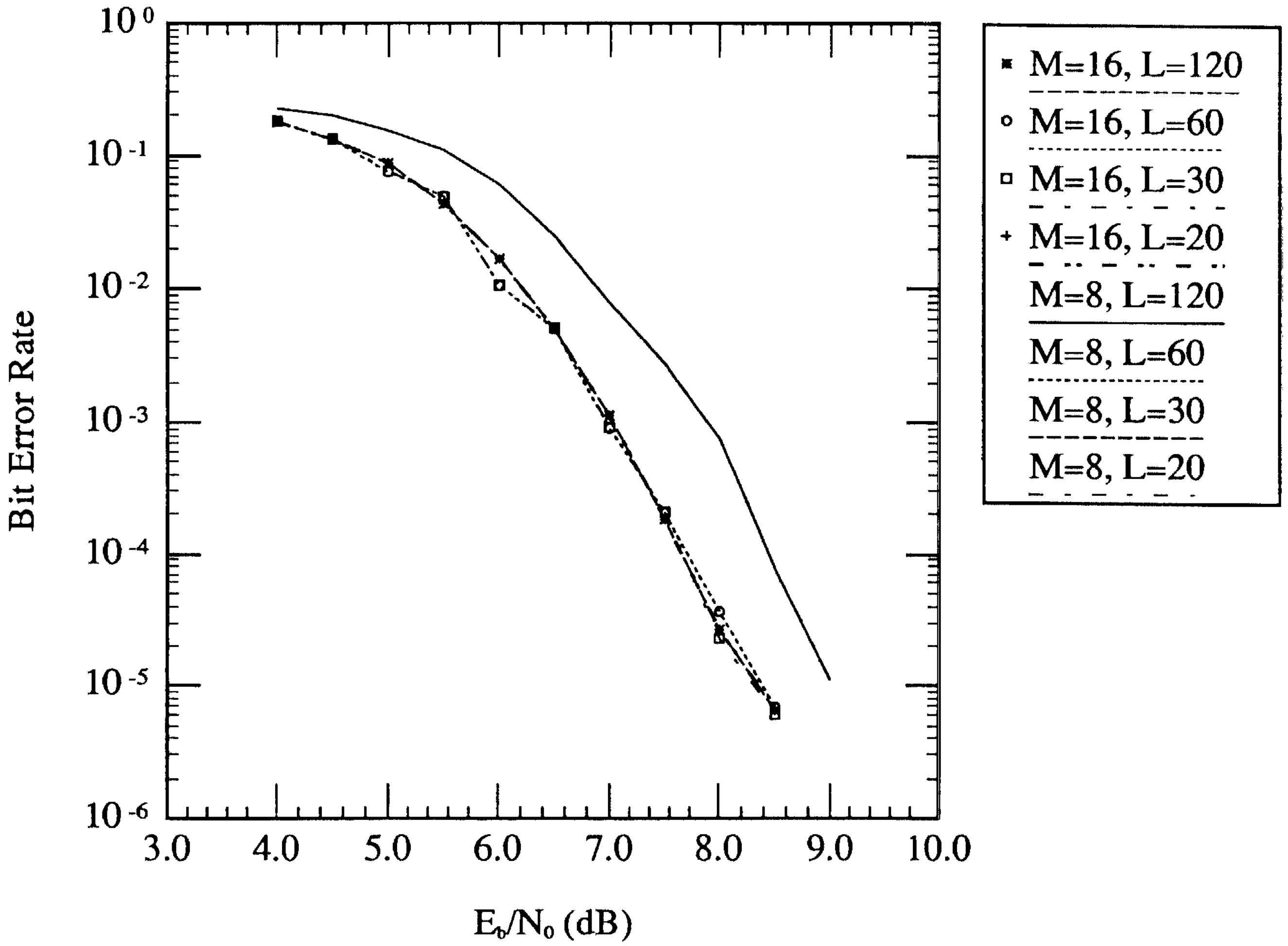


Figure 2: Simulation Results for the Nonlinear, 64 State, 16QAM Code.
Reduced State Space = 16

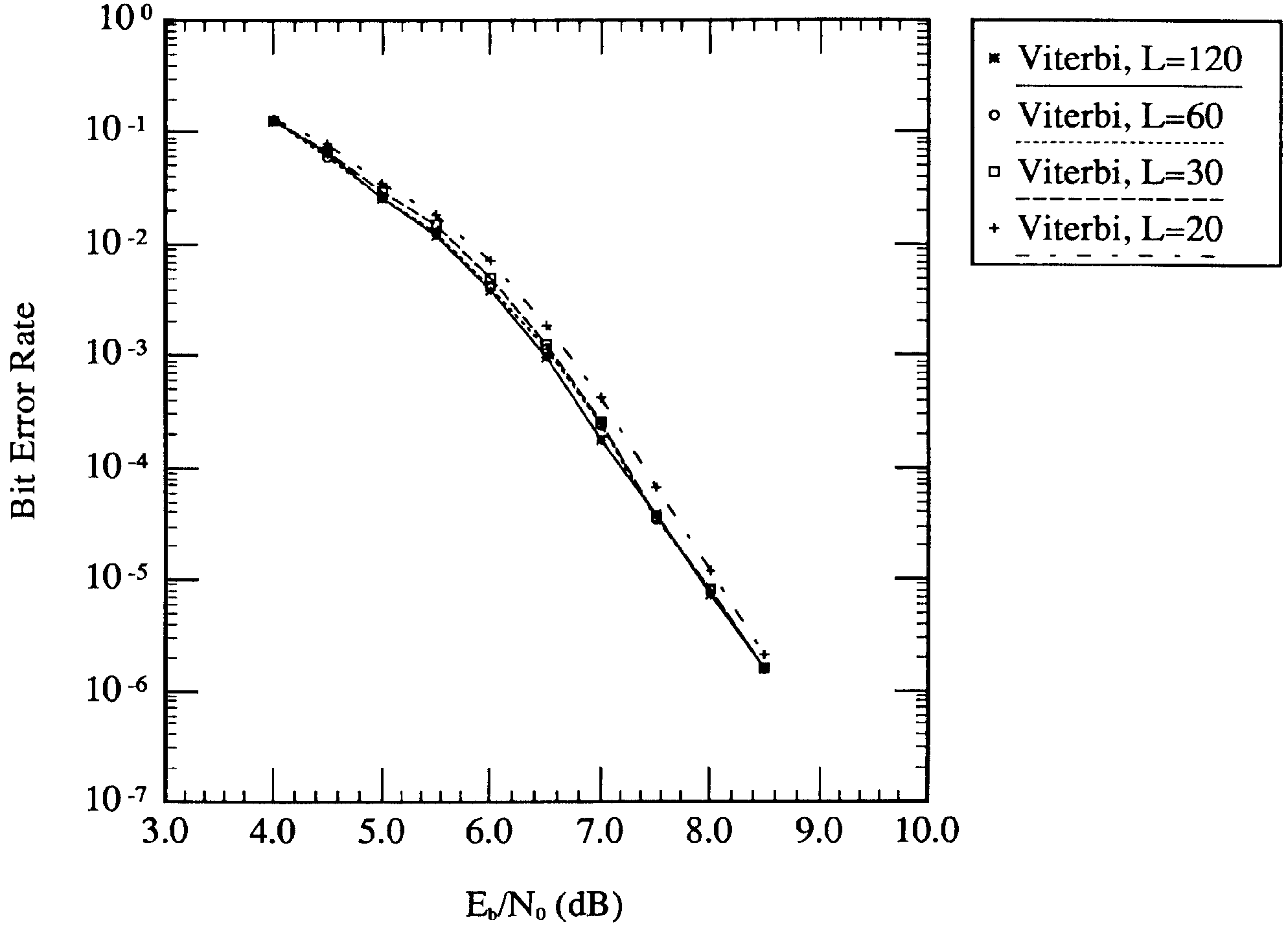


Figure 3: Simulation Results for the Nonlinear, 64 State, 16QAM Code.
64 State Viterbi Decoding

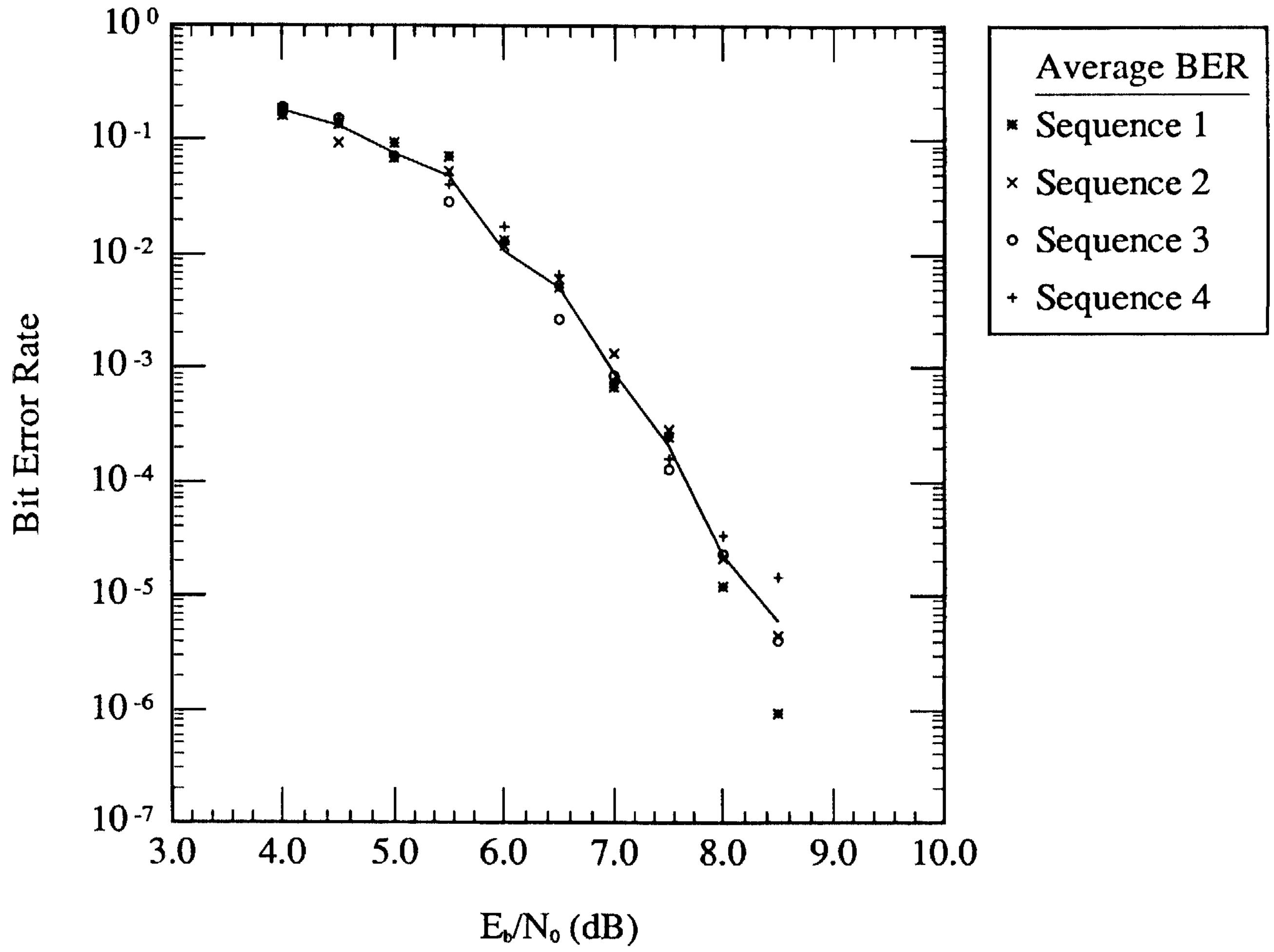


Figure 4: Simulation Results for the Nonlinear, 64 State, 16QAM Code.
Reduced State Space = 16

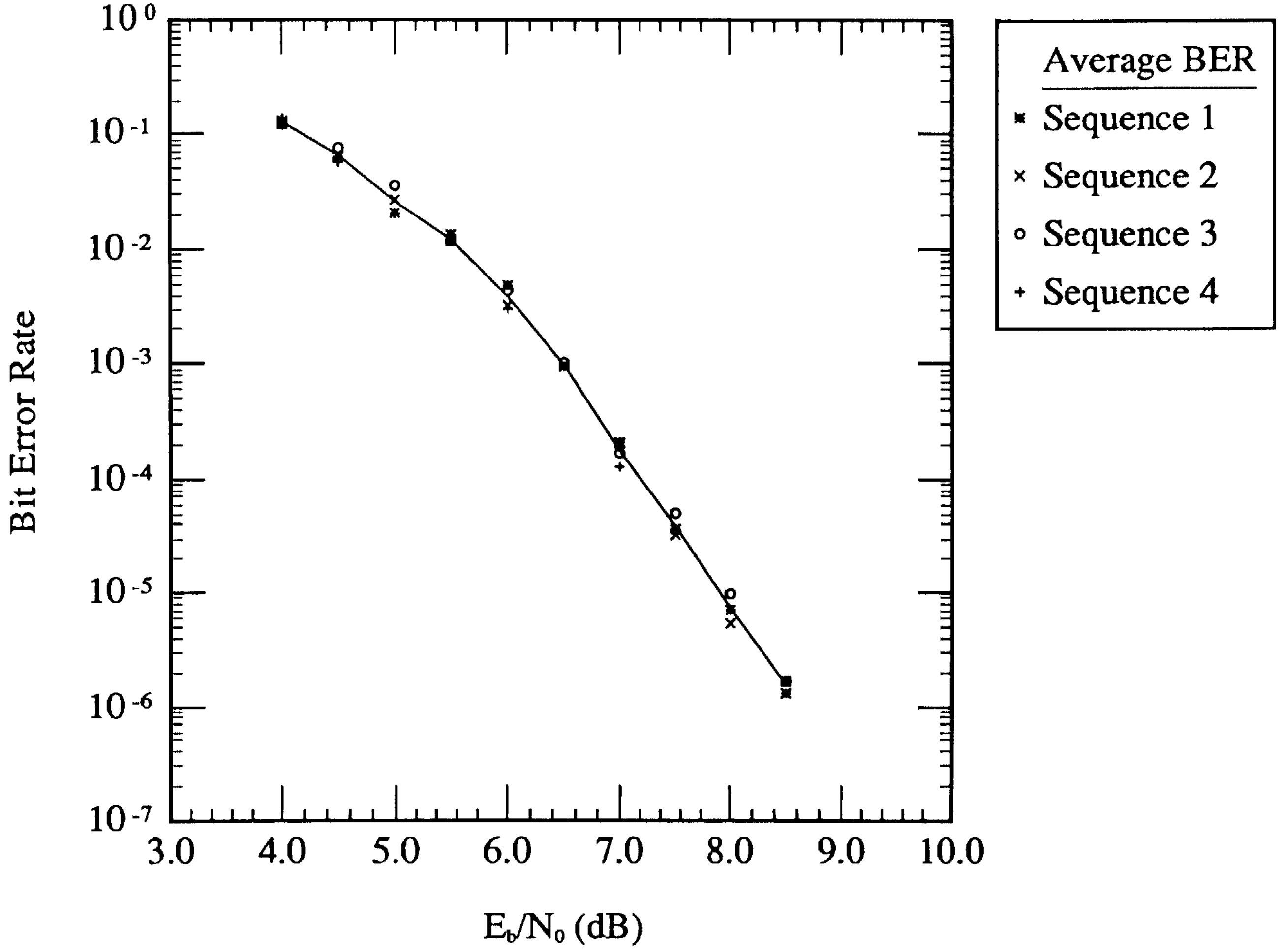


Figure 5: Simulation Results for the Nonlinear, 64 State, 16QAM Code. 64 State Viterbi Decoding